

Dr. Dobb's Journal of Software Tools

FOR THE PROFESSIONAL PROGRAMMER

BENCHMARKING C COMPILERS

The Joy of
Conciseness

Nearly
Perfect Trees

Generics
in Ada

Real-World
Data Types



TAKE THE NEXT LOGICAL STEP

\$89 Price

- Separate Compilation
- Native Code Generation
- Large Memory Model Support
- Multitasking
- Powerful Debugging Tools
- Comprehensive Module Library
- Available for the PC and the VAX

Move up to LOGITECH MODULA-2/86. Whether you're a single programmer or part of a team, with LOGITECH MODULA-2/86 you'll decrease your overall development cycle and produce more reliable, more maintainable code. Build your program using our extensive library modules, your own modules or those from a growing list of available third-party software vendors. If you're a Turbo Pascal user you can even take your existing code along with you with the help of our new Translator!

NEW & IMPROVED!

Turbo Pascal to Modula-2 Translator

\$49

Now it's even easier for Turbo users to step up to Modula-2/86. Our improved Translator changes your Turbo source code into Modula-2/86 source, solving all the incompatibilities, and translating the function calls of Turbo into Modula-2/86 procedures. Implements the complete Turbo library!

LOGITECH MODULA-2/86

\$89

Complete with Editor, Run Time System, Linker, 8087 Software Emulation, Binary Coded Decimal (BCD) Module, Logitech's comprehensive library, Utility to generate standard .EXE files. AND more!

- LOGITECH MODULA-2/86 with 8087 Support **\$129**
 - LOGITECH MODULA 2/86 PLUS **\$189**
- For machines with 512K of RAM. Takes advantage of larger memory to increase compilation speed by 50%.

Turbo Pascal is a registered trademark of Borland International.



MODULA-2/86

RUN TIME DEBUGGER

\$69

(Source level!)

The ultimate professional's tool! Display source code, data, procedure call chain and raw memory. Set break points, assign values to variables, pinpoint bugs in your source.

UTILITIES PACKAGE

\$49

Features a post-mortem debugger (PMD). If your program crashes at run time the PMD freezes the situation so you can pinpoint, in the source, the cause of the error and the status of the data. Also includes a disassembler, cross reference utility and version that allows conditional compilation.

LIBRARY SOURCES

\$99

Source code for our major library modules is now available for customization or emulation.

WINDOW PACKAGE

\$49

Now you can build true windowing into your Modula-2 code. Powerful, though only 15K in size. Features virtual screens, color support, overlapping windows and a variety of borders.

MAKE UTILITY

\$29

Automatically selects modules affected by code changes to minimize recompilation and relinking. Even figures out dependencies for you!

CROSS RUN TIME DEBUGGER AND ROM PACKAGE

\$199

Now available at an introductory price!

NEW, improved Turbo Pascal to Modula-2 Translator!

Call for information about our VAX/VMS version, Site License, University Discounts, Dealer & Distributor pricing.

To place an order call our special toll free number:

800-231-7717

In California:

800-552-8885

YES

I want to move up to LOGITECH MODULA-2/86!

Here's the configuration I'd like:

- | | |
|---|--------------|
| <input type="checkbox"/> Logitech Modula-2/86 | \$89 |
| <input type="checkbox"/> with 8087 support | \$129 |
| <input type="checkbox"/> Plus Package | \$189 |
| <input type="checkbox"/> Turbo to Modula Translator | \$49 |
| <input type="checkbox"/> Run Time Debugger | \$69 |
| <input type="checkbox"/> Utilities Package | \$49 |
| <input type="checkbox"/> Library Sources | \$99 |
| <input type="checkbox"/> Window Package | \$49 |
| <input type="checkbox"/> Make Utility | \$29 |
| <input type="checkbox"/> ROM Package | \$199 |

Total Enclosed \$ _____

☐ Visa ☐ Mastercard ☐ Check Enclosed

Card Number _____ Expiration Date _____

Signature _____

Name _____

Address _____

City, State _____

Zip _____ Phone _____



LOGITECH

Logitech, Inc.
805 Veterans Blvd.
Redwood City, CA 94063
Telephone 415-365-9852

For European pricing please contact:

Logitech SA
Box 32, CH-1143
Apples, Switzerland
Telephone 41-21-774545



**Those who insist on C compiler performance
are very big on Mark Williams.**

And the compiler is just part of our total C Programming System.

**NEW 3.0
ENHANCEMENTS**

These and other powerful utilities now *included* in the C Programming System:

- **make:** compiles only what's necessary from multiple modules, a powerful programming discipline
- **diff:** identifies differences between two files
- **m4:** macroprocessor expression editing and substitution
- **egrep:** extended pattern search
- **MicroEMACS:** full screen editor with source

COMPILER FEATURES

- Runs under MS-DOS
- Full Kernighan & Ritchie C with recent extensions including void and enum
- Register variables for fast, compact code
- Full UNIX™ compatibility and complete libraries
- Large and small memory models
- MS-DOS linker compatibility
- 8087 Support
- One-step compiling
- English error messages
- ROMable code
- Linker, assembler, archiver
- Extensive third party library support

csd C SOURCE DEBUGGER

- Debugs at C source level without assembly language
- Separate evaluation, source, program and history windows
- Can execute any C expression
- Capabilities of a C interpreter, but runs in real time
- Set trace points on any statement or variable

Mark Williams' C compiler has earned a place in some very big companies for some very good reasons: it proves the benchmarks right with the speed, code density, consistent performance and expert support required in professional development environments.

But a total development tool shouldn't stop with compiling. Or go on and on with extras that add up and up.

Only Mark Williams' C Programming Systems *includes* the csd C Source Debugger with true source level debugging to speed your programming job.

And only Mark Williams' new 3.0 version *includes* utilities like "make" to make quick work of even the largest projects.

From source code to final product, only one takes you all the way: Mark Williams' C Programming System. All for only \$495. **Ask about our 60-day money back guarantee when you call**

1-800-692-1700 to order today.*

You'll be big on the total C Programming system from Mark Williams, too.



1430 West Wrightwood
Chicago, Illinois 60614

The Peak of Performance

SCALE THE HEIGHTS OF PRODUCTIVITY

Sure, you've proven that in your hands a computer is a productive tool. But if you haven't teamed up with a SemiDisk you have heights yet to climb!

IT'S NO MERE RAMDISK

SemiDisk has been leading the way for Disk Emulators since their inception. If you've seen RAMdisks you know what it's like to load programs in an

instant, and read or write files without delay. Unlike alternatives, the SemiDisk offers up to 8 megabytes of instant-access storage while leaving your computer's main memory free for what it does best - computing!

KEEP A GRIP ON DATA

Go ahead, turn off your computer. Take a vacation. With the battery backup option, your valuable data will be there in the morning even if you aren't. You'll sleep better knowing not even a 5 hour blackout will sabotage your files.

SEMI

SemiDisk Systems, Inc.
P.O. Box GG, Beaverton, Oregon 97075

503-626-3104

NEW LOWER SEMIDISK PRICES THAT WON'T SNOW YOU UNDER

	512K	2Mbyte
IBM PC, XT, AT	\$495	\$995
Epson QX-10	\$595	\$995
S-100, SemiDisk II	\$799	\$1295
S-100, SemiDisk I	\$595	—
TRS-80 II, 12, 16	\$695	\$1295
Battery Backup Unit	\$130	\$130

Software drivers available for CP/M 80, MS-DOS, ZDOS, TurboDOS, and VALDOCS 2.

Circle Reader Inquiry Number 85

Dr. Dobb's Journal of Software Tools

FOR THE PROFESSIONAL PROGRAMMER

REVIEWS

**Professional
programmers
clock C
compilers** ▶

C COMPILERS: Benchmarking C Compilers **30**
by Richard Relp, Steve Hahn, Fred Viles
It's said that in the high-stakes poker palaces of San Jose, California, lowball games have been known to run for weeks, the players coming and going without affecting the game. Since we published our first extensive comparative review of C compilers for MS-DOS last year, several players have cashed in their chips and moved on, but new players—such as IBM—have arrived. The review team has again wielded its collection of surgical benchmarks, each designed to measure a particular aspect of compiler performance, and it's added a typical-performance benchmark, the dhrystone.

COLUMNS

**Nearly perfect
trees in C** ▶

C CHEST: An AVL Tree Database Package **20**
by Allen Holub
Allen presents a solution to the problems that arise when you insert a sorted list of keys into a binary tree.

16-BIT SOFTWARE TOOLBOX: Forth and the EMS **70**
by Ray Duncan
Ray's source code for the PC-Forth interface he introduced in July

**Ada, Modula-2,
and Pascal
extensions** ▶

**STRUCTURED PROGRAMMING: Generic Routines in
Ada and Modula-2, Pascal Iterators** **116**
Nimir Clement Shammis
Nimir shows how to get around the limitations of data typing. He describes routines that handle different-size arrays with the same data types and then explains routines that handle different data types. Nimir also takes a look at a new extended *for* loop in a Pascal compiler.

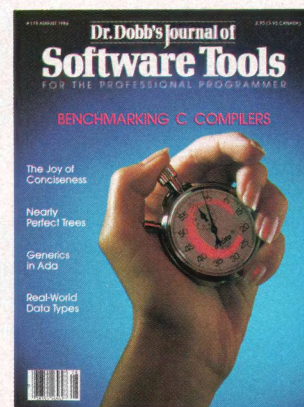
FORUM

**Flames from Old
Nick** ▶

EDITORIAL **6**
by Michael Swaine
RUNNING LIGHT **8**
by Nick Turner
LETTERS **10**
by you
CARTOON **10**
by Rand Renfro
VIEWPOINT: **16**
The Joy of Conciseness
by Allen Holub
DDJ ON LINE: **18**
Bluesky
by you
SWAINE'S FLAMES **128**
by Michael Swaine

PROGRAMMER'S SERVICES

DR. DOBB'S CATALOG: **73**
DDJ products—all in one place
OF INTEREST: **120**
Many new products of interest to programmers
ADVERTISER INDEX: **126**
Where to find those ads



About the Cover

Tom Upton did the cover photography. The hand belongs to Jill A. Meniketti.

This Issue

In our second annual review of C compilers for MS-DOS we improve and expand the process we used last August. This year, we present in-depth evaluations and comparisons of 17 compilers. Allen Holub continues his discussion of binary trees. In Structured Programming, Nimir Shammis shows how to use generic routines in Modula-2 and Ada to avoid reinventing the wheel for different data types.

Next Issue

The draftsman's spline, used to draw curves, is a rapidly disappearing tool. In our algorithm issue, Ian E. Ashdown develops a mathematical model of the drafting tool. Ian shows how to use this model to develop a program that can interpolate a smooth curve between a set of given points.

**YOUR
COMPUTER LANGUAGE
IS QUIETLY
BREEDING REAL BATS
IN YOUR
BELFRY.**



LANGUAGES THAT ARE CAUSING THE BIGGEST PROGRAMMING BACKLOG IN HISTORY ARE ALSO EATING NICE BIG HOLES IN OUR POCKETS.

Whether it's BASIC, COBOL, Pascal, "C", or a data base manager, you're being held back.

Held back because the language has frustrating limitations, and the programming environment isn't intuitive enough to keep track of what you're working on.

In the real world, there's pressure to do more impressive work, in less time, and for more clients.

We've been given some incredibly powerful hardware in recent times, but the languages aren't a whole lot better than they were 20 years ago.

So, whatever language you have chosen, by now you feel it's out to get you — because it is.

Sure, no language is perfect, but you have to wonder, "Am I getting all I deserve?"

And, like money, you'll never have enough.

Pretty dismal, huh?

We thought so, too.

So we did something about it.

We call it CLARION™.

You'll call it "incredible."

Distributed on 7 diskettes, CLARION consists of over 200,000 lines of code, taking 3+ years to hone to "world-class" performance.

With CLARION you can write, compile, run and debug complex applications in a New York afternoon.

Even if you're in Savannah.

It gives you the power and speed to create screens, windows and reports of such richness and clarity you would never attempt them with any other language.

Because *you* would have to write the code.

With CLARION you simply design the screens using our SCREENER utility and then CLARION writes the source code AND compiles it for you in seconds.

Likewise, you can use REPORTER to create reports.

Remember, only CLARION can recompile and display a screen or report layout for modification.

And with no time wasted.

All the power and facilities you need to write great programs, faster than you ever dreamed of.

Programs that are easy to use.
Programs that are a pleasure to write.

And to you that means true satisfaction.

You've coveted those nifty pop-up help windows some major applications feature. But you can't afford the time and energy it takes to write them into your programs.

That's the way it used to be.

So we fixed that, too.

CLARION's HELPER is an interactive utility that let's you design the most effective pop-up help screens that you can imagine. And they're "context sensitive," meaning you can have help for every field in your application.

Unlike the other micro languages, CLARION provides declarations, procedures, and functions to process dates, strings, screens, reports, indexed files, DOS files and memory tables.

Imagine making source program changes with the CLARION EDITOR. A single keystroke terminates the EDITOR, loads the COMPILER, compiles the program, loads the PROCESSOR and executes the program. It's that easy!

Our data management capabilities are phenomenal. CLARION files permit any number of composite keys which are updated dynamically.

A file may have as many keys as it needs. Each key may be composed of any fields in any order. And key files are updated whenever the value of the key changes.

Like SCREENER and REPORTER, CLARION's FILER utility also has a piece of the CLARION COMPILER. To create a new file, you name the Source Module. Then you name the Statement Label of a file structure within it.

FILER will also automatically rebuild existing files to match a changed file structure. It creates a new record for every existing record, copying the existing fields and initializing new ones.

Sounds pretty complicated, huh?

Not with CLARION's documentation and on-line help screens. If you are currently competent in BASIC, Pascal or "C" you can be writing CLARION applications in a day. In two days you won't believe the eloquence of your CLARION programs.

Okay, now for the best part of all. You can say it in CLARION for \$295.00—plus shipping and handling. All you need is an IBM® PC, XT, AT or true compatible, with 320 KB of memory, a hard disk drive, and a parallel port. And we'll allow a full 30 day evaluation period. If you're not satisfied with CLARION, simply return it in its original condition for a full refund.

If you're not quite ready to take advantage of this no-risk opportunity, ask for our detailed 16 page color brochure. It vividly illustrates the elegance of CLARION. Consider it a preview of programming in the fast lane.

Either way, the 800 call's a freebie.



SAY IT IN

CLARION™

Dept. A2ST/3

1-800-354-5444

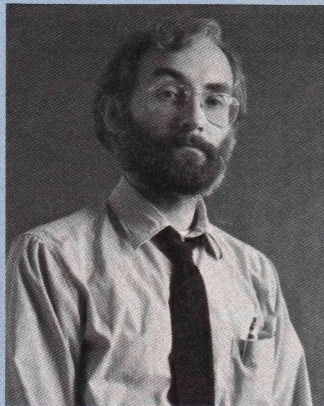


BARRINGTON SYSTEMS, INC. 150 EAST SAMPLE ROAD POMPANO BEACH, FLORIDA 33064 305/785-4555

IBM is a registered trademark of International Business Machines Corporation. CLARION™ is a trademark of Barrington Systems, Inc. ©1986 Barrington Systems

Circle no. 115 on reader service card.

EDITORIAL



If you were watching one particular recent broadcast of the film *Falcon and the Snowman* on HBO, you were probably surprised to see appear on your screen in the midst of the film the message, "Good evening from Captain Midnight. \$12.95 a month? No way. Showtime Movie Channel Beware."

Or maybe you weren't surprised. Wasn't it inevitable that satellite television would get its own Captain Crunch?

Apparently, Captain Midnight's beef is that HBO has begun scrambling its broadcasts in response to the increasing use of dish antennas that allow the user to receive satellite broadcasts such as HBO's without paying. The \$12.95 the Captain referred to is HBO's monthly charge for descrambler boxes.

The Showtime Movie Channel was threatened because Showtime is considering using a similar scrambling scheme.

What the Captain has done, according to a report in the British journal *New Scientist*, is prove that "it is technically possible for third parties to hijack a broadcast satellite and hold the cable industry to ransom." And if such third parties only operate in brief bursts, they may be extremely hard to track down.

Science fiction writer and telecommunications expert Arthur C. Clarke did some projections over 20 years ago and found that the amount of power available to the average individual, extrapolated from past figures, will go asymptotic before the year 2000. Because infinite power is meaningless, Clark concluded that we'd just have to wait and see how much power would be available to whom and what would be done with it.

It looks like the results are starting to come in.

This month's issue is dominated by a review of C compilers for MS-DOS. In all, fifteen companies submitted compilers for the detailed benchmarking. They are:

C Ware Corp., P.O. Box C, Sunnyvale, CA 94087; (408) 720-9696

Computer Innovations, 980 Shrewsbury Ave., Tinton Falls, NJ 07724; (201) 542-5920

Datalight, 11557 8th Ave. NE, Seattle, WA 98125; (206) 367-1803

Ecosoft Inc., 6413 N. College Ave., Indianapolis, IN 46220; (317) 255-6476

IBM, 1000 N.W. 51st St., Boca Raton, FL 33432; (305) 998-2000

Lattice, P.O. Box 3072, Glen Ellyn, IL 60138; (312) 858-7950

Manx Software Systems, P.O. Box 55, Shrewsbury, NJ 07701; (800) 221-0440
Mark Williams Co., 1430 W. Wrightwood Ave., Chicago, IL 60614; (312) 472-6659

MetaWare, 412 Liberty St., Santa Cruz, CA 95060; (408) 429-6382

Microsoft, 16011 N.E. 36th Way., P.O. Box 97017, Redmond, WA 98073-9717; (206) 882-8080

MixSoftware, 2116 E. Arapaho, Ste. 363, Richardson, TX 75081; (214) 783-6001

Software Toolworks, 15233 Ventura Blvd., Ste. 1118, Sherman Oaks, CA 91403; (818) 986-4885

Whitesmiths Ltd., 97 Lowell Rd., Concord, MA 02174; (800) 225-1030

Wizard Systems Software, 11 Willow Ct., Arlington, MA 02174; (617) 641-2379

WordTech Systems Inc., 21 Altarinda Rd., Orinda, CA 94563; (415) 254-0900

Michael Swaine

Michael Swaine
editor-in-chief

Dr. Dobb's Journal of Software Tools

Editorial

Editor-in-Chief Michael Swaine
Editor Nick Turner
Managing Editor Vince Leone
Assistant Editor Sara Noah Ruddy
Technical Editor Allen Holub
Contributing Editors Ray Duncan
Allen Holub
Michael Ham
Namir Shamma

Copy Editor Rhoda Simmons
Electronic Editor Levi Thomas

Production

Production Manager Bob Wynne
Art Director Michael Hollister
Assoc. Art Director Alida Hinton
Typesetter Jean Aring
Cover Artist Tom Upton

Circulation

Newsstand Sales Mgr. Stephanie Barber
Circulation Director Maureen Kaminski
Book Marketing Mgr. Jane Sharninghouse
Circulation Assistant Kathleen Shay

Administration

Finance Manager Treva Rafalski
Business Manager Betty Trickett
Accounts Payable Supv. Mayda Lopez-Quintana
Accts. Pay. Coordinator Kathy Robinson
Accounts Payable Asst. Karen Green
Accounts Receivable Mgr. Laura Di Lazzaro
Accts. Receivable Asst. Denise Giannini
Adm. Coordinator Kobi Morgan
Advertising Director
Robert Horton (415) 366-3600
Account Managers
Michele Beaty (317) 875-8093
Lisa Boudreau (415) 366-3600
Gary George (404) 897-1923
Michael Wiener (415) 366-3600
Cynthia Zuck (718) 499-9333
Promotions/Srves. Mgr. Anna Kittleson
Advertising Secretary Michelle A. Davie

M&T Publishing, Inc.

Chairman of the Board Otnar Weber
Director C.F. von Quadt
President and Publisher Laird Foshay

Dr. Dobb's Journal of Software Tools (USPS 307690) is published monthly by M&T Publishing, Inc., 501 Galveston Dr., Redwood City, CA 94063; (415) 366-3600. Second-class postage paid at Redwood City and at additional entry points.

Article Submissions: Send manuscripts and disk (with article and listings) to the Assistant Editor.

Address Correction Requested: Postmaster: Send Form 3579 to *Dr. Dobb's Journal*, P.O. Box 27809, San Diego, CA 92128. **ISSN 0888-3076**

Customer Service: For subscription problems call: outside CA (800) 321-3333; within CA (619) 485-9623 or 566-6947. For order problems call (415) 366-3600.

Subscription Rates: \$29.97 per year, U.S. Foreign rates: \$56.97, air; \$46.97, surface. Foreign subscriptions must be pre-paid in U.S. dollars, drawn on a U.S. bank. For foreign subscriptions, TELEX: 752-351.

Foreign Newsstand Distributor: Worldwide Media Service, Inc., 386 Park Ave. South, New York, NY 10016; (212) 686-1520 TELEX: 620430 (WUI).

Entire contents copyright © 1986 by M&T Publishing, Inc. unless otherwise noted on specific articles. All rights reserved.



People's Computer Company

Dr. Dobb's Journal of Software Tools is published by M&T Publishing, Inc. under license from People's Computer Company, 2682 Bishop Dr., Suite 107, San Ramon, CA 94583, a non-profit corporation.



The C for Microcomputers

PC-DOS, MS-DOS, CP/M-86, Macintosh, Amiga, Apple II, CP/M-80, Radio Shack, Commodore, XENIX, ROM, and Cross Development systems

MS-DOS, PC-DOS, CP/M-86, XENIX, 8086/80x86 ROM

Manx Aztec C86

"A compiler that has many strengths ... quite valuable for serious work"

Computer Language review, February 1985

Great Code: Manx Aztec C86 generates fast executing compact code. The benchmark results below are from a study conducted by Manx. The Dhrystone benchmark (CACM 10/84 27:10 p1018) measures performance for a systems software instruction mix. The results are without register variables. With register variables, Manx, Microsoft, and Mark Williams run proportionately faster, Lattice and Computer Innovations show no improvement.

	Execution Time	Code Size	Compile/Link Time
Dhrystone Benchmark			
Manx Aztec C86 3.3	34 secs	5,760	93 secs
Microsoft C 3.0	34 secs	7,146	119 secs
Optimized C86 2.20J	53 secs	11,009	172 secs
Mark Williams 2.0	56 secs	12,980	113 secs
Lattice 2.14	89 secs	20,404	117 secs

Great Features: Manx Aztec C86 is bundled with a powerful array of well documented productivity tools, library routines and features.

Optimized C compiler	Symbolic Debugger
AS86 Macro Assembler	LN86 Overlay Linker
80186/80286 Support	Librarian
8087/80287 Sensing Lib	Profiler
Extensive UNIX Library	DOS, Screen, & Graphics Lib
Large Memory Model	Intel Object Option
Z (vi) Source Editor -c	CP/M-86 Library -c
ROM Support Package -c	INTEL HEX Utility -c
Library Source Code -c	Mixed memory models -c
MAKE, DIFF, and GREP -c	Source Debugger -c
One year of updates -c	CP/M-86 Library -c

Manx offers two commercial development systems, Aztec C86-c and Aztec C86-d. Items marked -c are special features of the Aztec C86-c system.

Aztec C86-c Commercial System	\$499
Aztec C86-d Developer's System	\$299
Aztec C86-p Personal System	\$199
Aztec C86-a Apprentice System	\$49

All systems are upgradable by paying the difference in price plus \$10.

Third Party Software: There are a number of high quality support packages for Manx Aztec C86 for screen management, graphics, database management, and software development.

C-tree \$395	Greenleaf \$185
PHACT \$250	PC-lint \$98
HALO \$250	Amber Windows \$59
PRE-C \$395	Windows for C \$195
WindScreen \$149	FirstTime \$295
SunScreen \$99	C Util Lib \$185
PANEL \$295	Plink-86 \$395

MACINTOSH, AMIGA, XENIX, CP/M-68K, 68k ROM

Manx Aztec C68k

"Library handling is very flexible ... documentation is excellent ... the shell a pleasure to work in ... blows away the competition for pure compile speed ... an excellent effort."

Computer Language review, April 1985

Aztec C68k is the most widely used commercial C compiler for the Macintosh. Its quality, performance, and completeness place Manx Aztec C68k in a position beyond comparison. It is available in several upgradable versions.

Optimized C	Creates Clickable Applications
Macro Assembler	Mouse Enhanced SHELL
Overlay Linker	Easy Access to Mac Toolbox
Resource Compiler	UNIX Library Functions
Debuggers	Terminal Emulator (Source)
Librarian	Clear Detailed Documentation
Source Editor	C-Stuff Library
MacRam Disk -c	UniTools (vi, make, diff, grep) -c
Library Source -c	One Year of Updates -c

Items marked -c are available only in the Manx Aztec C86-c system. Other features are in both the Aztec C86-d and Aztec C86-c systems.

Aztec C68k-c Commercial System	\$499
Aztec C68d-d Developer's System	\$299
Aztec C68k-p Personal System	\$199
C-tree database (source)	\$399
AMIGA, CP/M-68k, 68k UNIX	call

Apple II, Commodore, 65xx, 65C02 ROM

Manx Aztec C65

"The AZTEC C system is one of the finest software packages I have seen"

NIBBLE review, July 1984

A vast amount of business, consumer, and educational software is implemented in Manx Aztec C65. The quality and comprehensiveness of this system is competitive with 16 bit C systems. The system includes a full optimized C compiler, 6502 assembler, linkage editor, UNIX library, screen and graphics libraries, shell, and much more. The Apple II version runs under DOS 3.3, and ProDOS. Cross versions are available.

The Aztec C65-c/128 Commodore system runs under the C128 CP/M environment and generates programs for the C64, C128, and CP/M environments. Call for prices and availability of Apprentice, Personal and Developer versions for the Commodore 64 and 128 machines.

Aztec C65-c ProDOS & DOS 3.3	\$399
Aztec C65-d Apple DOS 3.3	\$199
Aztec C65-p Apple Personal system	\$99
Aztec C65-a for learning C	\$49
Aztec C65-c/128 C64, C128, CP/M	\$399

Distribution of Manx Aztec C

In the USA, Manx Software Systems is the sole and exclusive distributor of Aztec C. Any telephone or mail order sales other than through Manx are unauthorized.

Manx Cross Development Systems

Cross developed programs are edited, compiled, assembled, and linked on one machine (the HOST) and transferred to another machine (the TARGET) for execution. This method is useful where the target machine is slower or more limited than the HOST, Manx cross compilers are used heavily to develop software for business, consumer, scientific, industrial, research, and educational applications.

HOSTS: VAX UNIX (\$3000), PDP-11 UNIX (\$2000), MS-DOS (\$750), CP/M (\$750), MACINTOSH (\$750), CP/M-68k (\$750), XENIX (\$750).

TARGETS: MS-DOS, CP/M-86, Macintosh, CP/M-68k, CP/M-80, TRS-80 3 & 4, Apple II, Commodore C64, 8086/80x86 ROM, 68xxx ROM, 8080/8085/Z80 ROM, 65xx ROM.

The first TARGET is included in the price of the HOST system. Additional TARGETS are \$300 to \$500 (non VAX) or \$1000 (VAX).

Call Manx for information on cross development to the 68000, 65816, Amiga, C128, CP/M-68K, VRTX, and others.

CP/M, Radio Shack, 8080/8085/Z80 ROM

Manx Aztec CII

"I've had a lot of experience with different C compilers, but the Aztec C80 Compiler and Professional Development System is the best I've seen."

80-Micro, December, 1984, John B. Harrell III

Aztec C II-c (CP/M & ROM)	\$349
Aztec C II-d (CP/M)	\$199
C-tree database (source)	\$399
Aztec C80-c (TRS-80 3 & 4)	\$299
Aztec C80-d (TRS-80 3 & 4)	\$199

How To Become an Aztec C User

To become an Aztec C user call 1-800-221-0440 or call 1-800-832-9273 (800-TEC WARE). In NJ or outside the USA call 201-530-7997. Orders can also be telexed to 4995812.

Payment can be by check, COD, American Express, VISA, Master Card, or Net 30 to qualified customers.

Orders can also be mailed to Manx Software Systems, Box 55, Shrewsbury, NJ 07701.

How To Get More Information

To get more information on Manx Aztec C and related products, call 1-800-221-0440, or 201-530-7997, or write to Manx Software Systems.

30 Day Guarantee

Any Manx Aztec C development system can be returned within 30 days for a refund if it fails to meet your needs. The only restrictions are that the original purchase must be directly from Manx, shipped within the USA, and the package must be in resalable condition. Returned items must be received by Manx within 30 days. A small restocking fee may be required.

Discounts

There are special discounts available to professors, students, and consultants. A discount is also available on a "trade in" basis for users of competing systems. Call for information.

MANX

To order or for information call:

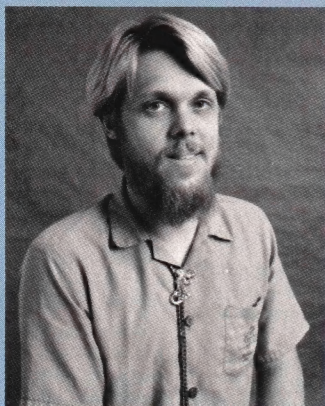
800-221-0440

UNIX is a registered TM of Bell Laboratories. Lattice TM Lattice Inc. C-tree TM Faircom, Inc. PHACT TM PHACT ASSOC. CI Optimizing C86 TM Computer Innovations, Inc. MACINTOSH, APPLE TM APPLE, INC. Pli-C, PLINK 86, TM PHOENIX, HALO TM Media Cybernetics, Inc. C-tree, PC-lint TM GIMPLE Software, WindScreen, SunScreen TM Sunlec, PANEL TM Roundhill Computer Systems Ltd., WINDOWS FOR C TM Creative Solutions, XENIX, MS TM MICROSOFT INC., CP/M TM DRI, AMIGA, C64, C128 TM COMMODORE INT.

Circle no. 108 on reader service card.

RUNNING LIGHT

A last-minute addition to this month's C compiler review: Mark Williams Company tells us that it has released Version 4.0 of its compiler. The new version has been substantially improved over Version 3.0.12, which is included in our review. Improvements include the ability to generate ROMable code, larger memory models, 80286 support, and the ability to link with the Microsoft and Lattice libraries.



and never need debugging of any sort. Yet I can walk into almost any programming shop and find coders who won't write a bug-free 20-line function in Pascal on the first try. These are people who know how to program—they just don't bother to do it right the first time.

I invite you to respond. This is a difficult and emotional issue. Let's get it out in the open where it can be solved.

This month's hint for writers has to do with "puff pieces"—articles that are little more than (usually glowing) descriptions of products available from the company for which the author works. Editorial ethics (and the focus of the magazine) require that we restrict articles to programming techniques, rather than glorifying specific products. Here's how to turn a puff piece into a useful article: First, don't even mention the product until the end, where you can safely say that the techniques demonstrated in the article are used in your product. Spend most of the article explaining techniques that will be useful to the reader whether or not he or she owns your product. Most important, include a self-contained sample program (which does not use or require your product) that demonstrates clearly how the reader can profit from your advice. Be sure to explain in the article what the demonstration program does and how it does it. Remember that *DDJ* is not a textbook. Our readers are often bored by reiterations of well-known techniques. Your material should be new, technically sound, and interesting to advanced programmers.

Nick Turner

Nick Turner
editor

I've been hearing about a "software gap" that allegedly threatens to become a major barrier to continued progress in computer programming. An article by John Paul Newport Jr. in the April 28 issue of *Fortune* magazine revealed some interesting information. In 1984, for example, data processing departments at 125 surveyed companies took an average of 27 months to deliver programs requested by other departments. The article mentions a study that found that 75 percent of the programs requested within companies are never used either because they are never completed or because they are outdated when they're delivered. What's going on?

Some people think that the problem is related to the size and complexity of the programs we're building these days. I disagree—it's simpler than that.

The problem is the simple sloppiness of many of today's "professional" programmers. It's not that most programmers don't know about structured design or self-documenting code. Rather, they just don't seem to care. I have known programmers who could write entire operating system kernels (hundreds of lines of source code) in one pass, in assembly code, that run perfectly the first time

ARCHIVES

Seeing C in '83

"The buzz is that C is the answer to our prayers, an expressive notation that compiles to efficient code."—*D.E. Cortesi, DDJ, November 1983.*

"I'm switching from a language that I deeply like (Pascal) to one that I'm uncomfortable with (C) simply because it is a better choice."—*Peter Norton, PC, September 1983.*

"C . . . I don't expect it ever to become a highly popular language."—*Jerry Pournelle, Byte, August 1983.*

FNE Well Grounded

"In a few months, AT&T will surely notice that the Unix PC [built by Convergent Technologies] is not selling. Let us assume that AT&T succeeds in stopping production after only 50,000 Unix PCs have been built. Of that 50,000 about 10,000 will have been sold. AT&T will purchase 8,500 itself, the recently divested operating companies, many of which have not yet wised up, will buy 1,483 units, and the rest of the world will buy 17."—*Hal Hardenbergh (a.k.a. FNE), dTack Grounded, June 1985.*

"Last year, Convergent shipped some 40,000 to 50,000 of the Unix PC units. But AT&T has only sold 10,000 . . . and many of those were sold to AT&T's own divisions. AT&T has said that the Unix PC . . . is 'ahead of its time.'"—*Brenton R. Schlender, Wall Street Journal, May 30, 1986.*

10 Years Ago in DDJ

"This bus structure as defined by MITS has become a 'de facto' standard."—*D. Denney and J. Broom, DDJ, August 1976.*

"The time for floppies is just about now. Gary Kildall is . . . hoping to be able to offer a controller in the neighborhood of \$350."—*Jim Warren, DDJ, August 1976.*

"One of the features of NCC '77 will be presentation of papers pertaining to personal computing. A milestone for personal computing!"—*Harold A. Mauch, DDJ, August 1976.*

"Although these routines are for the 6502 . . . [they could be modified] for most of the traditional microprocessors relatively easily. [They] were done by Steve Wozniak."—*Jim Warren, DDJ, August 1976.*

"There are some tentative rumors being passed about concerning a Computer Faire . . . Let us know if the prospect . . . interests you."—*Jim Warren, DDJ, August 1976.*

" . . . the 8080 sets the parity flag on both logical and arithmetic operations. Since Bill Gate's [sic] Altair BASIC uses this 8080 eccentricity . . . Altair BASIC will not run on a Z80."—*Jim Warren, DDJ, August 1976.*

"Three groups [have] boards or microcomputers based on Zilog's hot new m-p, the Z80: the Digital Group, TTL, and Cro-MemCo."—*Jim Warren, DDJ, August 1976.*

DR. DOBB'S JOURNAL of
COMPUTER
Calisthenics & Orthodontia
Running Light Without Overbyte

DATALIGHT C DELIVERS PERFORMANCE

DATALIGHT C \$60 ■ THE DEVELOPER'S KIT \$99

The **DATALIGHT C Compiler** is a performer through and through. From the UNIX System 5 language with the latest ANSI extensions (prototyping), to the top compile speed, with the understandable error interface, on to the tight/fast code — the Performer shines. Performance comes in two sizes: THE **DATALIGHT C Compiler**, and the **DEVELOPER'S KIT**.

DATALIGHT C provides you with a full range of features, like full 8087 and software floating point, a UNIX-style **MAKE** program, one-step compile/link, and UNIX-like tools in source form. You also get automatic .COM file generation, MS-DOS compatible object files, third-party library/debugger support, and much more.

The **DEVELOPER'S KIT** provides the extra features required by the serious programmer. The **DEVELOPER'S KIT** allows you to build programs as BIG as the memory in your PC. You can also tailor your applications to your special PC or ROM-based needs using the start-up and library source provided.

And what do our users think? They love us! In fact, the ones who

own higher-priced compilers love us the most because they know what **PERFORMANCE** really is!

"This is a sharp compiler! . . . what is impressive is that DATALIGHT not only stole the compile time show completely, but had the fastest Fibonacci executable time and had excellent object file sizes to boot!"

Chris Skelly

COMPUTER LANGUAGE

"I have and actively use Microsoft C version 3.0, Mark Williams C, and Lattice version 3.0, in addition to your compiler. Of all the compilers that I have used yours really stands out as the best package."

Matthew Brandt

TANGENT TECHNOLOGIES

"Of all the MS-DOS C compilers available, we have chosen DATALIGHT for all our development."

Keven Smith Drew Gilesen

TRAVELING SOFTWARE

So why wait? You can have the Performer, a 30-day money-back guarantee, and the call is on us. So order now!

DATALIGHT C (Ver 2.10)

- Full UNIX System 5 C Compiler with ANSI extensions.
- Fast/tight code.
- 8087 & software floating point.
- Full UNIX compatible library.
- **MAKE** program with macros, dependency checking, and MS-DOS internal commands.
- **DLC** one-step compile/link program.
- Tools in source (diff, cat, pr, wc, rm).
- Powerful utilities in source form.
- Compatible with MS-DOS linker.

DEVELOPER'S KIT (Ver 2.10)

- All features of **DATALIGHT C**!
- Third-party library support.
- Multiple memory model support
 - Small 64k code 64k data
 - Data 64k code 1Meg data
 - Program 1Megcode 64k data
 - Large 1Megcode 1Meg data
- Complete **SOURCE CODE** for libraries.
- Complete **SOURCE CODE** for start-up routine.
- ROMable code.

MS-DOS is a trademark of Microsoft.
UNIX is a trademark of Bell Labs.
Lattice C is a trademark of Lattice Inc.

Datalight

P.O. BOX 82441
KENMORE, WA 98028
(206) 367-1803

ORDER NOW! 30-DAY MONEY-BACK GUARANTEE.

YES, I WANT THE PERFORMER!!!

By phone call 1-800-628-2828 — Ext. 571 (Orders only)
Technical Information (206) 367-1803

I want _____ copies of **DATALIGHT C** (\$60)

I want _____ copies of the **DEVELOPER'S KIT** (\$99)

Add \$5 for shipping in US/\$15 outside US. C.O.D. (add \$2.50)

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

CARD # _____

EXPIRATION DATE _____

P.O. # _____ (attach copy)



LETTERS

**Lost at C**

Dear DDJ,

I have been an avid supporter of DDJ for years. It has always been a good source of programming ideas but I find it less and less useful because of the increasing emphasis and use of C.

I have no inherent objection to C—its only problem is that it is essentially unreadable to users of other languages. By printing listings in C, you limit their utility—for example, I would dearly have liked to be able to use some of the algorithms given in the article by Joe Marasco in the March 1986 issue. It is a topic of great importance to me, but they were unintelligible.

Several years ago, I became convinced that I ought to move from FORTRAN into the world of structured languages. The two obvious candidates were C and Pascal. I spent a few months reading about both of them and using them. My conclusion was that there were no intrinsic advantages of one over the other, but even for a neophyte such as myself, Pascal was so much easier to read. To me, the common criticism that it is wordy is a validation of its readability. I decided to use it from then on—a decision I've never regretted.

I'm a scientist first and a programmer last, however. Over the years, my recol-

lection of C has dimmed, and I can no longer recall the meaning of those funny ++ and other symbols. Hence, my complaint to you.

I don't suggest that programmers constrain themselves to Pascal. I do suggest that C is particularly unreadable, however, and I would like to ask you to urge authors to describe algorithms in a way that would allow non-C users to make use of them. Of course, I have no objection to listings in C (as well, but not alone) because I'm sure that many find them useful.

J. A. Koehler
Institute of Space and
Atmospheric Studies
Univ. of Saskatchewan
Saskatchewan, Canada

C Chest

Dear DDJ,

I have some comments about some of the things Allen Holub has said about his MS-DOS shell and about Microsoft C. In addition, I'm in the process of hacking at this shell, and I'd like to explain some of the enhancements I've added and intend to add.

First of all, I've been using Microsoft C, Version 3.0, for quite a while, and I tend to agree with most of Allen's comments and criticisms of it. I especially agree with his assessment of Microsoft's support policy. I, too, have found that they are generally uncooperative about giving support to people who ask questions that go deeper than those that a relatively high-level user

would ask. The firm seems to not care about giving this support to anyone who buys less than a hundred or so copies of its software. This infuriates me.

With regard to their Version 3.0 C compiler, Allen seemed to overlook a useful feature (that's not hard to do given the organization of the manual). There is a function called `_setargv()`, which every program calls to parse its command-line arguments. By putting Allen's `reargv()` function inside a routine with this name and then replacing the `_setargv()` function with this newly created one in the standard library the user can take advantage of the command-line argument expansion scheme without explicitly calling `reargv()`. This is almost as good as putting the call to `reargv()` into the root module.

Previous to getting Allen's shell, I had already written my own `_setargv()` routine, which expanded my command-line arguments in a similar way as `reargv()` does. So all I have to do is insert a call to `reargv()` at the top of my existing routine and exit if it works. If it doesn't, that means that the `CMDLINE` environment variable wasn't found. I can then fairly safely assume that the shell isn't running and go ahead and do my own expansion.

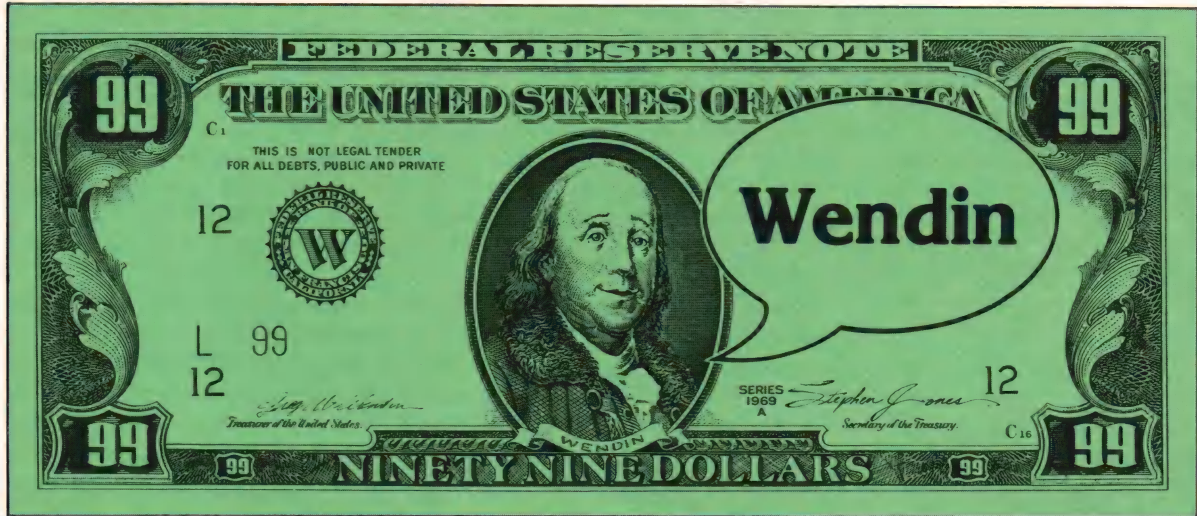
Lloyd Zusman
Master Byte Software
127 Wilder Ave.
Los Gatos, CA 95030

Allen Holub replies:

The `_setargv()` function is mentioned (once) in the manual, but no details are given about how it works. The routine is called from the start-up (or root) mod-



MONEY TALKS.



Are You Listening?

A lot of people mistakenly assume that the more a piece of software costs, the better it is.

And a lot of software companies take advantage of that.

At Wendin we think quality software can and should be affordable. That's why all our products are priced at \$99.

Take our Personal Operating Systems developed for the IBM PC and true compatibles. These include **PCUNIX™** and **PCVMS™**, both written with our powerful software construction set, the **Operating System Toolbox™**.

Similar to AT&T's UNIX operating system, PCUNIX has the most popular UNIX features, including more than 59 utilities. And we're adding new features all the time.

PCVMS is our version of the versatile VAX/VMS operating system, which duplicates nearly all the system services and many of the commands found on the popular mainframe original.

Because PCUNIX and PCVMS were developed with the Operating System Toolbox, programs written on either system

can access over 80 enhanced system services built into the Operating System Toolbox kernel. And like all operating systems built with the Operating System Toolbox, these multitasking, multiuser systems use the MS-DOS file system and run well behaved MS-DOS programs.

In addition to our Personal Operating Systems, we've developed one of the simplest and most powerful text editors on the market. **XTC®**. Combining a multitasking macro language with multiple linkable windows and buffers, XTC outperforms everything in its class.

It also costs less than anything in its class. And it comes with complete source code.

All our products do.

Of course, if you're one of those people for whom money is no object, all of this may not interest you. But for the rest of you, we'll say it one more time.

Wendin. \$99

Thanks for listening.

WENDIN®

BOX 266
CHENEY, WA 99004

© Copyright 1986 Wendin, Inc.

The people who make quality
software tools affordable.

MS is a trademark of Microsoft; PC-DOS is a trademark of IBM; UNIX is a trademark of AT&T; VAX/VMS is a registered trademark of Digital Equipment Corporation

Wendin and XTC are registered trademarks of Wendin, Inc. PCUNIX, PCVMS, Operating System Toolbox, and Personal Operating System are trademarks of Wendin, Inc.

ORDER HOTLINE

(509) 235-8088

(MON.-FRI., 8-5 PACIFIC TIME)



DEALER INQUIRIES WELCOME

Foreign orders inquire about shipping. Domestic orders add \$5.00/1st item, \$1.00 each additional item for shipping handling, and insurance. Washington residents add 7.8% sales tax.

LETTERS

(continued from page 10)

ule to initialize *argv* and *argc*. I called Microsoft and was told that it takes no arguments but must set the global variables `__argv` and `__argc` to the values that will be passed to *main*(). This is all fine and dandy, but when I tried to access the environment [with *getenv*()] I received an error message. As it appears that Mr. Zusman has gotten it working, I seem to have given up too easily.

Let Your Fingers Do the Talking

Dear DDJ,

I read the May 1986 interview with Jef Raskin with interest—also, the Edlin article—but the tops this month was your Viewpoint on voice input and output.

You missed one thing, however. The big shortcoming of voice out is speed. A man just cannot listen as fast as he can read. And if a person did not need speed he would not use a computer. I once got the idea that I would develop a system to get information out of a digital system via audio, which would be as fast as by visual means. The project failed miserably. I learned a lot about the shortcomings of the auditory system. That was all. My conclusion was that the ear-voice system was developed in unison and that when you tried to work either with a computer you're headed for trouble. Voice systems can be used in applications in which the eye must be otherwise engaged. That's on the output side. I never worked on the input side.

My company does a couple of things on the input side that might interest you. First, we enter code on a code sheet using hex

characters. We then read that code column with a photoreader, which can be built for about \$40. Then we also use a Combo Keyboard, which is ten keys placed so they fall naturally under the fingertips. The ASCII code can be generated directly by pressing the keys in combinations, much as a piano is played. Ten keys permit the generation of 1,023 characters. We use only eight of the keys to input code, which cuts us to 256 characters. We also use a one-hander with six keys, which inputs 63 characters in basic mode. We pull a trick or two to input 191. The chief advantage of the keyboards is that you never need to move your eyes to the board.

R. O. Whitaker

Computer Compatible
Inst. Co.

4719 Squire Dr.

Indianapolis, IN 46241

The Right to Optimize

Dear DDJ,

I've been intrigued with DDJ's continuing saga of integer square roots. Because most of the recent debate (The Right to Assemble by Richard Campbell, March 1986) has centered on whether the 68000 or 32032 is best suited for this task, I thought you might appreciate seeing how we poor slobs shackled to 8086s handle the problem. Actually, integer square roots are fairly important to me because I work with digitized gamma-ray spectra where estimates of error are intimately tied to counting statistics. The 8087 will blaze through square roots, but I can't assume my programs will be run on machines with 8087s, so I've been on the lookout for fast 8086 routines.

Listing One, page 81, is

written as a function for DeSmet C and uses DeSmet's *asm88* mnemonics; code for Intel and Microsoft assemblers should be very similar. As with Campbell's 320XX code, the underlying principle is Newton's method, and half the game is picking an appropriate initial estimate. Like the 68000, the 8086 is sluggish on divides (approximately 150-160 clocks each), so you might expect Campbell's 320XX code to vastly outperform the 8086. An 8-MHz 8086, however, averages only 172 milliseconds per 32-bit square root (based on "in vivo" timings, to use Campbell's phrase), which seems reasonably competitive with Campbell's result of 183 microseconds for a 6-MHz 16032. Note, Campbell states that his 320XX function took an average of only 7 shifts, but that's because he restricted his test program to integers between 0 and 60,000; for the full range of 32-bit numbers, the average would be 15 shifts, and the 16032 would be substantially slower. The 80286 does divides using temporary registers (22 clocks) and should be much faster than the 8086, taking less than 100 microseconds. I've tried other methods to get a better estimate before entering the *refine* loop, including faster versions of the *guess1*, *guess2* convergence method used by Campbell; oddly, these better initial estimates only speed up the 8086 by about 10 percent and can actually slow down the 80286. On the 80286, the extra logic is slower than extra divides.

I initially wrote an 8086 version of Jim Cathey's 68000 code (May 1985, DDJ), but it was comparatively slow, averaging about 250 microseconds; I've en-

closed the 8086 version of this bit-shifting method [Listing Two, page 81], and to say the least, it is opaque compared to the 68000 code. Adding up the execution clocks for the 8086 suggests that the routine should average only 150 microseconds at 8 MHz; however, short 8086 instructions such as *shl ax,1* (2 clocks) execute faster than they can be fetched, and the processor is constantly stopping to refill the prefetch queue. Hence, Campbell's advice about using "in vivo" timings proves very appropriate. In addition, the paucity of registers in the 8086 means that even *sp* has to be used to avoid storing intermediate results in memory. The 80286 has a much faster fetch rate and would tend to keep the queue filled, except after branching. The need to use all the registers would be relaxed in the 80286 because memory references can be almost as fast as register-register operations. For statistical decision making, I rarely need to calculate accurate square roots, and I use a function that returns an approximation (correct to within 10 percent) in 30 microseconds. Still, I'd appreciate hearing about faster 8086 routines.

Without doubt, the 320XX and 68000 are much easier to program than are the 8086 and 80286, but it is often surprising how well the Intel chips do when a modest amount of thought goes into the code. I disagree with Campbell's bias against using low-level, bit- and byte-oriented tricks; sometimes these methods are the fastest and cleanest, and who is to say which approach is more "rational"? Cathey's bit-shifting 68000 code reflects the square root method most



USE THE BRAINS YOUR IBM WASN'T BORN WITH.

Right at your fingertips in CompuServe's IBM® Forums.

In the **IBM New Users Forum** you'll swap ideas with other new PC users, learn to use Forum features, and pose even basic questions to PC experts.

Our **IBM Junior Forum** gives PCjr® users a reliable source for tips on software, hardware, telecommunications, games and other interests.

In the **IBM Software Forum** you'll trade tips with other IBM PC and AT users on utility software, word processing, DOS and other operating systems.

Visit the **IBM Communications Forum** for advice on the features and compatibility of communications software and hardware, PC Bulletin Boards, micro-mainframe interfaces and more.

The **IBM Hardware Forum** addresses hardware topics of all types, plus product updates and announcements.

Easy access to free software.

- Download first-rate, non-commercial, user-supported software and utility programs.
- Take advantage of CompuServe's inexpensive weeknight and weekend rates (when Forums are most active, and standard online charges are just 10¢ a minute).
- Go online in most major metropolitan areas with a local phone call.
- And receive a **\$25.00 Introductory Usage Credit** with purchase of your CompuServe Subscription Kit.

Information you simply can't find anywhere else.

Use the **Forum Message Board** to send and receive electronic messages, and pose specific questions to other IBM and compatible owners.

Join ongoing, real-time discussions in a **Forum Conference**.

Search our unparalleled **Forum Data Libraries** for free software, user tips, transcripts of online conferences and more.

Enjoy other useful services like:

- **Popular Computer Magazines**—electronic

editions, for your reading pleasure. Including *Dr. Dobbs's Journal* and *Computer Language*.

• **Other CompuServe Forums**—supporting LOTUS® products like *Symphony™* and *1-2-3.™* Borland International®, Ashton-Tate®, Digital Research®, MicroPro®, Microsoft® and other software. Also Pascal, Basic, C, Fortran, Assembly and other programming languages.

All you need is your IBM or IBM-compatible computer and a modem ... or almost any other computer.

To buy your Subscription Kit, see your nearest computer dealer. Suggested retail price is \$39.95. To receive our free brochure, or to order direct, call 800-848-8199 (in Ohio, call 614-457-0802). If you're already a CompuServe subscriber, type GO IBMNET (the IBM Users Network) at any prompt to see what you've been missing.

CompuServe®

Information Services, P.O. Box 20212
5000 Arlington Centre Blvd., Columbus, Ohio 43220

800-848-8199

In Ohio, Call 614-457-0802
An H&R Block Company

LETTERS

(continued from page 12)

people learn in seventh-grade algebra and probably seems much more intuitive to those who stayed awake in class.

H. W. Stockman
6807 Prairie Rd. NE, #405
Albuquerque, NM 87109

Dear DDJ,

Will we ever see the last word on square roots? Although I do agree with Mr. Campbell [The Right to Assemble, March 1986] that the bit-level mashing that is necessary for 8-bit processors is undesirable, it is not lightly dispensed with as I have found (not to mention how interesting it is to see how much better it comes out on the better processor). The square root routine that I actually use is much better, although quite a bit larger [Listing Three, page 82]. My real routine is broken into three parts—a part for arguments no larger than a single word, a part for arguments larger than a single word (with two of the loops unrolled so that a quick word-oriented loop can be used when there is no danger of overflow), and a special routine that handles particularly small arguments that is used when it would be quicker than the normal word routine. It should be noted that this program yields correct results over the entire range of arguments from 0 to \$FFFFFFF, which I believe cannot be said for Mr. Campbell's routine if the NS32000 processor handles \$FFFFFFF ÷ \$FFFF like the 68000 does (overflow, with no result). This may be of no significance to most users, but it was of importance to me because of the way I originally used the routine.

What I find interesting about the shift-based rou-

tine is that it is of no more complexity than a divide routine, and it does not depend upon converging to a solution. As a hardware-oriented kind of guy, I can see how when microcoded it could calculate a square root in about the same time as a divide. This simplicity is what attracted me to the algorithm in the first place. To me, this algorithm has the same sort of elegance as Bresenham's algorithm for line drawing and CORDIC for coordinate rotation.

As my C compiler spits out a subroutine call for the division rather than using the *DIVU* instruction, I hand-coded a version of Mr. Campbell's routine for the 68000 [Listing Four, page 85]. For cycle-shavers even more diligent than myself, I think there is still some room for trimming, but I lost interest in trying further. This version also attempts to cover the full range of arguments, although I cannot vouch that it does it successfully. I spot-checked a few of the large troublemakers, and it did those correctly.

The results of comparing these two routines were most instructive. Mr. Campbell is correct in his feeling that the shift-based routine is slower than Newton's method if there is a hardware divide of the requisite size available. I have found, however, (on the 68000 at least) that if you only need the square root of a word-sized number, the shift-based routine runs faster than Newton's method! My version of Mr. Campbell's routine takes an average of 854 cycles to calculate each of the first 65,535 square roots (that's seven seconds total to you and me). My shift-based routine takes an average of 610 cycles to do the same thing (5 seconds). For the first 500,000 roots,

the Newton's method routine averages 992 cycles (62 seconds) vs. 1,104 cycles (69 seconds) for my routine. This is not too much worse than Newton's method. What is interesting to note is that both the best- and worst-case timings for the shift-based routine are better than Newton's method, but the average favors Newton's method. This is almost entirely because of the quality of the initial guess calculated by Mr. Campbell's routine. Newton's method is lousy as a root-guesser unless the initial guess is close—then it is excellent.

As far as instruction pipelines and their effect on cycle counting goes, the 68000 also has a one-word look ahead, but the cycle counts listed by Motorola supposedly include this time, so accurate counts are possible. This is not possible on the 68020, though. Because Motorola hedges a little when listing divide clock times, the counts listed for Newton's method may be off a little. I think a version of the shifting routine for the NS32000 may be more efficiently coded using the add/addc style of rotating two bits out of the argument into the trial register. This might reduce the discrepancy between Mr. Campbell's two versions and my two versions. I cannot say, though, as I am unfamiliar with the NS32000 at that level of detail.

I must say that I am pleased with the amount of thought that my submission has spawned. Perhaps one of the poor souls afflicted with the 8086 could whomp up the two routines and see how they compare.

Jim Cathey
ISC Systems Corp.
TAF-C8
Spokane, WA 99220

Faster Random Numbers

Dear DDJ,

The pseudorandom number generator given in the November 1985 16-Bit Software Toolbox may be too slow for some applications. The enclosed routine was inspired by the article "A Fast Method of Generating Digital Random Numbers" by Rader, Rabiner, and Schafer in the November 1970, *Bell System Technical Journal* and is essentially shift register feedback.

Two data registers of the 68000 should be dedicated to the routine because the overhead of subroutine calling is too slow. These two registers contain the seed numbers, and their concatenation constitutes a 64-bit shift register. The following four instructions constitute the generator:

EXG	D6,D7
ROL.L	#3,D7
SUBQ.W	#7,D7
EOR.W	D6,D7

The *EXG* instruction acts as a 32-bit rotate; the *ROL* instruction acts as a partial rotate; the *SUBQ* instruction avoids the pitfall of all zeros; the *EOR* instruction mixes the bits. Each time the sequence is performed, a new seed is generated and the low byte or low word of D7 is available as a random number. At 8 megahertz, the time required is 3.5 microseconds, so it is almost a hundred times faster than the earlier routine.

Lawrence Mertz
287 Fairfield Ct.
Palo Alto, CA 94306

DDJ

(Listings begin on page 81.)

HAUPPAUGE

Is Getting A Fast Reputation.



HAUPPAUGE started earning a fast reputation with their 87 Math Pak, the combination of an 87 chip and 87 Software Pak that's been accelerating PC math since 1982.

Next came their racy 287 FAST/5, a math coprocessor module with its own 5MHz clock, speeding up PC/AT math by 25%. (Pictured above.)

Now, Hauppauge Unveils the 287 FAST/8A...

Our newest math coprocessor for the PC/AT, the 287 FAST/8A moves out at 8MHz—doubling the speed of each floating point math operation. The FAST/8A accelerates AutoCad, 1-2-3, Symphony, Turbo Pascal, Framework and more. The FAST/8A also runs in PC/AT compatibles including the Compaq Deskpro 286, Sperry PC/IT and TI Business-Pro computers.

...And the 87 Software Pak Version 6.0

Designed to steal the heart of programmers, the 87 Software Pak supports IBM's BASIC Compiler 1.0 and 2.0, and Microsoft's QuickBASIC, executing math-intensive programs up to 20 times faster! The 87 Software Pak also performs FFT's and Matrix operations. For example, a PC (or PC/XT) with an 87 Chip and 87 Software Pak can perform a 512-point complex FFT in just 1.1 seconds. What's more, a PC/AT with a FAST/8A inverts a 25 by 25 element matrix in under 1 second.

Circle no. 274 on reader service card.

Hauppauge

(Pronounced "Ha-pog")

HAUPPAUGE Math Coprocessors

287 FAST/8A 8MHz math coprocessor for PC/AT and compatibles...\$379

287 FAST/5 5MHz math coprocessor for PC/AT and compatibles...\$249

287 Chip PC/AT math coprocessor—runs at 4MHz in PC/AT...\$219

87 Chip Math coprocessor for IBM PC, PC/XT and compatibles...\$129

87-2 Chip Math coprocessor for 8MHz PC compatibles...\$195

HAUPPAUGE Math Coprocessor Paks

87 Math Pak V.6.0 87 chip and math coprocessor software support for IBM BASIC Compiler 1.0, 2.0 and Microsoft's QuickBASIC. Plus, Matrix and FFT support, one year of free updates, complete source code and "8087 Applications and Programming".....\$279

87 Software Pak V.6.0 Math coprocessor software support as in the 87 Math Pak, but without 87 chip.....\$180

With any Hauppauge math coprocessor.....\$150

Recalc+ Math coprocessor support for 1-2-3 version 1A...\$ 95

With any Hauppauge math coprocessor.....\$ 49

HFT+ Complete Hayes Fourier Transform Package.....\$125

With any Hauppauge math coprocessor.....\$ 79

The 287 FAST/8 Doubles Your PC/AT's Math Speed!

Help your PC/AT get a fast reputation with Hauppauge's new 287 FAST/8A. Call today, or contact your local computer dealer to learn more about Hauppauge's racy product line. And ask for "87 Q & A," our free booklet on math coprocessors.

Hauppauge Computer Works, Inc.

358 Veterans Memorial Highway, Suite MSI,
Commack, New York, USA 11725 • 516-360-3827
Available at your local computer dealer

We acknowledge the following registered trademarks: 1-2-3 and Symphony: Lotus Development Corporation; AutoCad: Autodesk, Inc.; IBM: PC, PC/AT, PC/XT; Compaq: Deskpro; Sperry: PC/IT; Texas Instruments: Business Pro; Ashton-Tate: Framework.

VIEWPOINT

Not too long ago Byte ran an article entitled "Easy C" by Pete Orlin and John Heath (vol. 11, no. 5, May 1985). Because the misconceptions in the article are representative of a class of mistakes often made by people who discuss the language, I feel obliged to comment.

The article opined that the "confusing notation and the unfriendly look" of C made it difficult to learn and difficult to program. The authors' solution was to use the macro preprocessor to change the look of the language, to make a program "more understandable." For example, they suggested changing:

```
switch( *s )
{
  case 'x':
    except = 1;
    break;
  case 'n':
    number = 1;
    break;
  default:
    printf("...");
    argc = 0;
    break
}
```

into:

```
CASE( *s )
  CASEOF( 'x' )
    except = 1;
  ENDCOF
  CASEOF( 'n' )
    number = 1;
  ENDCOF
```

by Allen Holub

```
DEFCASE
  printf("...");
  argc = 0
ENDCOF
ENDCASE
```

or changing:

```
( c >= 'a' && c <= 'z' )
into:
```

```
( c GE 'a' AND c, LE 'x' )
```

by using macros such as:

```
#define GE >=
#define LE <=
#define AND &&
#define INC ++
#define FOR(e) { for (e)
  { #define ENDFOR; } }
#define CASE(e) { switch (e) {
  #define ENDCASE; } }
```

You get the idea.

I'm reminded of COBOL. I don't know any programmers who actually like the COBOL language (though several of them are used to it at this point and so wouldn't change without an argument). The main objection is that everything is done with words, so you can't skim through a program. It's difficult to see a program's structure at a glance, and you actually have to read every word for the program to make sense. As a consequence, COBOL programs are difficult to write, they take a long time to debug, and they are hard to maintain.

COBOL was originally sold to managers because, in theory, it would give them control over their programmers. *Control* is the operative word here. Because the language uses an English-like syntax, the theory was that managers would be able to read the programs and thus keep tabs on what the programmers were doing. There'd be no "confusing" symbols such as the *NE* or *GT* used in FORTRAN. There'd be no terse constructs (such

as the *DO* loop) that the manager couldn't understand. Because the manager could read the code, he or she could tell when a programmer was goofing off. How could it possibly take that long to write a simple computer program anyway? Somebody must be cheating.

Unfortunately, computer programs just do take a long time to write, and COBOL programs are no more readable than any other kind of program. Using English words instead of other symbols (words are after all just symbols) makes a program no more comprehensible to non-programmers. By the time these managers found out that a computer program is not a business plan, it was too late. Enough code had been written in COBOL that moving to another language wasn't financially feasible. As a consequence a large body of hard-to-maintain COBOL programs are in use today. Nobody wins. The managers still can't read the code, and the programmers have a harder time writing it.

It bothers me to see essentially this same argument applied to C. I agree that C code can be terse at times. Nonetheless, Mr. Orlin and Mr. Heath are confusing familiarity with readability. Just because you know another programming language doesn't mean that everyone who programs in C is also going to know the same language. Because I don't know the language into which the authors are trying to transform C, I find their transformed programs incomprehensible. Is *<=* really less un-

derstandable than *LE*? Is *switch* less understandable than *CASE*?

To carry the argument to the logical extreme, what if everyone did what the authors suggest? Because everyone is familiar with a different set of languages, everyone would make up a different set of macros to make their code "more understandable." All programmers would then have their own personal programming language. You wouldn't be able to read anyone's code but your own without having to learn a new programming language first. By the same token, if you never bothered to learn the correct C syntax, you wouldn't be able to read a program that was written in standard C. A sorry state of affairs indeed.

There are other problems here, too. For example, a lot of books tell you to:

```
#define FALSE 0
#define TRUE 1
```

so that you can say:

```
if( subr() == TRUE )
```

Because any nonzero value is true in C, the *if* statement can evaluate incorrectly when *subr()* returns a perfectly reasonable true value that doesn't happen to be 1. So, potential errors have been introduced.

The real difficulty here stems from a misunderstanding of the actual problem, and it's a misunderstanding that extends to topics other than C. C is a difficult language to learn, as are many branches of computer science. The difficulty is not caused by the symbols that the language

uses, however, but rather by the fundamental structure of the language itself: the ways that pointers are used and so forth. Going further, the structure of the language is in large part determined by the structure of a computer. So the problem isn't really the language but rather the amount of general knowledge of computers that you must acquire before C begins to make sense. You have to understand a lot about how computers work before you can even understand what the C operators do, much less how to use them. It's not that the $>$ and $>>$ operators look alike, but that many novice programmers don't know what an arithmetic right shift is. In the C classes that I teach at U.C. Berkeley,

I've noticed that assembly-language programmers usually have little difficulty learning about pointers because they understand indirect addressing.

C was developed in order to write operating systems, a task not attempted by most novice programmers. It was never intended to be anyone's first language, nor was it intended to be intelligible to unsophisticated programmers. In fact, the very things that would make C intelligible to beginners would also make the language useless for its intended purpose: systems programming. Try to write an operating system in BASIC sometime. On the other hand, languages such as Pascal and its cousins were designed to teach you how to program and

are excellent for this purpose. If you're new to the programming business, you should be learning Pascal not C.

A well-written C program is a joy to read, once the syntax is familiar. (How long can it take to learn that \leq means less than or equal?) The best way to learn C is to learn Pascal, assembly, and a little bit about computer architecture and hardware design. No amount of beating your head against the language will help you if you don't have the background. No one can write a good C program unless they have the background. If you're trying to learn C, you're wasting your time changing the way the language looks. C is not easy, it's not for beginners, and no amount of

wishful thinking will make it so. There are no shortcuts. On the other hand, the effort spent learning the language, and learning the things you need to learn the language, can't help but make you a better programmer.

DDJ


Unified Field Theory

Finally. A unified, easy to learn, flexible and powerful system to define and control all your input and output fields in a single, elegant command.

Introducing C-scape: A revolution in C.

Blow away those tedious hours spent laboring over menus and input screens. **C-scape** takes fields, menus, prompts, and text and unifies them into a single, powerful command. Beginning with completely customizable attributes, validation, types, colors, scrolling, virtual windows, even the way the cursor moves from field to field and character to character, **C-scape** offers a multi-dimensional library that will grow to fill your potential and make you a star. You'll see your development time contract and your productivity expand. Your universe will never be the same.

Another Key to Freedom, from

Oakland Group, Inc. 

675 Massachusetts Avenue, Cambridge, MA 02139

And using **Dan Bricklin's Demo Program**, you can bang out prototypes in a flash. Then, at the speed of light, **C-scape** will turn those demo screens into actual, tangible C code. Compile and link with the **C-scape** library and presto: Your creation is done!

30-day guarantee: Try **C-scape** for 30 days and see how it unifies simplicity, depth, and power in a single command. Following registration you'll get **full source code**, updates, and support. **C-scape** requires no run-time license and no royalties. We're developers, too. We know your needs.

\$149.00 C-scape (Lattice/Microsoft 3.0; others call)

\$219.00 C-scape with Dan Bricklin's Demo Program

Please add \$3.00 for shipping. Massachusetts orders please include 5% sales tax.



For orders and information, call:

**617-491-7311, or
800-233-3733**

Circle no. 227 on reader service card.

DDJ ONLINE

We are now having weekly real-time conferences on the DDJ Forum. The conferences will take place on the DDJ Forum CO nference channel 30 and give readers a chance to "chat" with DDJ editors, writers, and special guests. To participate, please check the DDJ Forum CO bulletin for conference times and read the CO help file in DLO.

Bluesky

#: 1321 S1/C Chest

Fm: Darryl 75206,3074

To: Bob 76003,102

Well . . . I think the Mac is basically an idea before its time. Now before you and every other Mac fan scream at me for this, let me explain.

As I stated before, I like the ideas embodied in the Mac. I don't like the Mac as it now exists (emphasis on *now*). It really needs (at the very least) a dedicated graphics chip to do hardware line drawing, area fills, and so on and some slots. Having much of the graphics done in hardware would do wonders to speed things up. It needs slots because, in my opinion, slots are used to fix problems that the manufacturer (Apple, in this case) did not anticipate. If the Mac had slots, adding to it new and useful capabilities would be much cheaper than it is now. Adding 4 megabytes RAM could cost only \$150 + cost of RAM. Adding more serial and parallel ports would be much cheaper. (Using the SCSI requires the use of more logic and, naturally, adds more to the price of the final product.) The Mac could be used for cheap data acquisition if it had slots. Data acquisition? If the Mac had slots, people would be using it for many

purposes for which it is not designed (I've heard the Apple II is used for videotape editing), and the Mac would be bought by many more people.

However, such a machine as I've just described would probably cost \$3,000-\$4,000, placing it out of the price range of many potential buyers. This is why I said it's before its time.

Fm: Bob

To: Darryl

A slotted Mac is in development, it was held up by the original design philosophy espoused by Steve Jobs and others, but it has no such restrictions on its appearance now and merely awaits development.

Fm: Darryl

But is it supported by Apple? If it's not, there won't be many compilers written to take advantage of it.

The Sieve program is really a poor benchmark for the 68881. The Sieve is basically an integer-only program that doesn't exercise the capabilities of the 68881.

Fm: Bob

Yeah, you're right about the Sieve as a test of the 68881. I didn't have any other figures at my fingertips, however. Duane Maxwell 74075,1666 is a principal in a company that provides an 020/881 upgrade and can provide you with more details.

Fm: Mike 70010,147

Hey, the full-blown Macintosh sounds like a good time, but I'd set a few bucks aside for a Fresnel lens to put in front of the screen. 1K×1K on such a small screen could get tough. Why not beg for an Amiga

with the same resolution and some Unix . . . wait a minute, the Amiga has a 68000 in it like the Integral . . . I wonder if . . . er, uh, excuse me, I gotta run.

Fm: Darryl

Oh, of course! If I had a screen resolution of 1K×1K, I'd want a 14- to 19-inch screen, not a puny 9-inch. Actually, I think one of the Mac's problems is that the 68000 is doing too many things. Having some dedicated graphics chips to handle the graphics would probably go a long way to solving some of the speed problems. Also, does the Mac have dedicated disk-controller chips? If it's anything like the way the disk I/O is done on the Apple II, then this is another area that could be improved.

Actually, if HP would transform the Integral from a portable to a trans-portable (say Compaq size) and add an HD and some slots, then it would be a very nice machine. Of course you're right; any machine like that would cost about \$10,000.

Fm: Mike

Early last year I was on the road all the time training our customers how to use our controller. I got pretty tired hanging out in Holiday Inn lounges and thought a portable would be a fun thing. OK, so I drop the bucks, get the IPC, and I haven't been on the road since, at least no overnights. The computer just sits on the desk at home . . . skip the 19-inch monitor, maybe I'll go for a projection TV.

Fm: Darryl

<grin> Better still how about getting a cellular

phone and a modem—that way you can be (almost) anywhere (in the city) and still be able to access CIS or whatever <grin>.

Fm: Mike

OK, but you still need a screen of some sort. Most LCDs would be miserable in a car with the limited viewing angle. ELs wouldn't be so bad, but they get dark (not sure why) in bright sunlight. Plasma displays would be dangerous in an accident—going through a windshield would be one thing, but going through a high-voltage plasma display would be something else entirely. Can you imagine DDJ Forum bounced off the windshield in a heads-up cockpit display? Neither can I.

Fm: Darryl

Now there's a thought. Install a heads-up display; a cellular phone connected to a modem that is connected to your CMOS MicroVAX; a speech synthesis/recognition board; and a CD-ROM to hold maps of the city, state, and U.S. That way you can talk to CIS as you navigate (using the heads-up display). Better still, add a few GaAs Crays, some sensors (sonar, radar detector, TV cameras), and a few mechanical linkages here and there to control your car. Now, all you'll have to do is get in the car and tell it where to go. It will do all the rest—even speed when there are no police around. (That's another reason why you have the radar detector and vision system.)

Of course, I won't be responsible if some joker takes a D-Stat gun to the car.

Fm: Mike

Of course, when the Cray

mobile needed a fill-up, you'd have to find yourself a GaAs station. (Sorry.)

Fm: Larry 75046,606
Actually, I hear the Sun 2/50, a diskless Sun that can have a tape unit added on later, can be had for approximately \$7,000 now . . . almost free <grin>. [This is not to be confused with] a \$30,000 Sun workstation (which I use at work each day...).

Fm: Mike
Hey Darryl, who is this guy! Sitting in front of a Sun workstation every day . . . and getting paid for it! Some guys get all the luck! Then again, maybe because it is "work," it's lost some of its glamour. I remember, when I was a kid, I thought it would be great to "work" with computers when I grew up.

Now I'm (almost) grown-up, and sometimes I just don't feel like going into the office for another day of C.R.T. (cathode-ray tan). Whaddya say Larry, has it lost its glamour? Is it just another day at the office? Tell me (please) you don't enjoy it!

Fm: Larry
Well, it is interesting. We are alpha-testing (pre-alpha?) the latest Sun workstation enhancements; so early I cannot even tell you. Other than the normal hassles of working in that early of an environment, through an OEM, and without adequate training, it isn't so bad.

What I like most about the whole thing is the keyboard! This is really strange. The machines are OK, but we have been using a dumb, vt100-emulat-

ing, terminal-mode type setup for development, with a keyboard that is much better than an IBM mishmash but strange nonetheless. Now I try a Sun keyboard, and it is a dream. What I find strange is that, with a setup like this, Sun doesn't have a very good setup with function keys. They are there (10-20, I really don't remember right now), but in the SunTools environment—a desktop where each window is a pseudo-TTY into a Unix 4.2BSD process—I can find no doc on how to take advantage of the function keys. Rather strange for a \$50,000 unit!

Fm: Mike
It doesn't sound like you hate your job all that much. Ultra-new equipment is a gas to work with, exploring new frontiers

and all that. Your comment on the keyboard hits home. It's funny how one's appreciation for a machine, be it a \$5,000 Integral or \$50,000 Sun workstation, is determined by the friendliness of a \$100 keyboard. My brother, a serious pianist, used to complain about piano keyboards in the same way, and I could never quite fully understand. Now I do, the IBM keyboard may be bad, but I'd trade the PC/AT keyboard in the office for this sticky, rubber HP keyboard in a minute.

DDJ

Vote for your favorite feature/
article.
Circle Reader Service No. 1.

Multi-User Version 3.0 **C-INDEX**TM Data Management for C Language Product Development

- Fast B+Tree Access
- Multi-Key Indexing
- Variable Length Records
- Record Locking
- Automatic Buffering
- Portable Source Code

"The C-Index package is the finest C language library that I have worked with. The design of the package is excellent in terms of functionality, ease of use and portability".

Dale Chamberlain, Professional Drug Systems

"I admire the work you have done with C-Index. Had I known about your product when I was designing dBASE III, I would have certainly used it."

Wayne Ratliff, Author of dBASE II and dBASE III†

C-Index/Plus	\$395	Full Source Code
C-Index/File	\$99	Object Code Only
Evaluation Package	\$25	Includes Manual

Trio Systems

2210 Wilshire Blvd. Suite 289 Santa Monica, CA 90403
213/394-0796

†dBASE II and dBASE III are trademarks of Ashton-Tate

Circle no. 150 on reader service card.

C for yourself!

A full year for only \$18.

Think about it, a full year of technical and useful information about C. **The C Journal** provides programming information for any machine - IBM PC[™], UNIX[™] -based, Macintosh[™], or CP/M[™] - micro, mini or mainframe. Look forward to each issue for:

- NEW in-depth reviews and feature articles - C compilers, editors, interpreters, function libraries and books.
- NEW efficiency hints and tips.
- NEW interviews with C experts.
- NEW news and rumors from the ANSI standards committee and industry.

Subscribe today to the *only* magazine that is dedicated specifically to C - **The C Journal**.

Please send check or money order for \$18 (cover price \$28) to:

InfoPro Systems

3108 Route 10, Denville, NJ 07834
Call TOLL FREE (800) 628-2828 ext. 849
(for charge card orders only)

Please add \$9 for overseas mail and \$6 for Canadian subscriptions.



THE C JOURNAL[™]

Trademarks - IBM PC: IBM Corp.; UNIX: AT&T Bell Labs; Macintosh: Apple Computer Corp.; CP/M: Digital Research Inc.; **The C Journal**: InfoPro Systems.

Circle no. 194 on reader service card.

An AVL Tree Database Package

AVL Trees

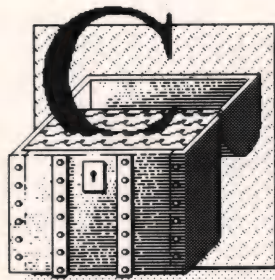
One of the problems with binary trees is the degraded case. When you insert a sorted list of keys into a binary tree, you get a linear linked list. Not only does it take an inordinate amount of stack to traverse this list but also the search time to find a key at the end of the list is pretty awful (statistically you'd be better off searching a randomly ordered linear list). One solution to this problem is the AVL balanced tree, the topic of this month's column.

A few terms first: The *depth* (or *height*) of a tree is the number of links that have to be traversed to get from a tree's root to the node that is farthest from the root. A tree with only one node in it is of depth 0. A perfectly balanced tree is one in which the distance from the root to all leaves is the same (by *leaf* I mean a node with no descendants). Obviously, the number of nodes in a perfectly balanced tree has to be a power of 2 less 1 (1, 3, 7, 15, 31, and so on). Also the worst-case search time in a perfectly balanced tree is $\log_2(N)$, where N is the number of nodes rounded up to the nearest power of 2. A perfectly balanced tree has the property that, for any node, the heights of the two subtrees are the same.

AVL trees (named after their inventors Adelson-Velski and Landis) are almost perfectly balanced trees. They have the property that, for any node in the tree, the heights of the two sub-

by Allen Holub

trees will differ by at most 1. This will give you a perfectly balanced tree more often than not, though there are AVL trees in which the maximum difference in depth between leaves is 2. Figure 1, right, for example, shows a worst-case balanced AVL tree. There's a difference in depth of 2 between



node 0 and node 12, but from the perspective of the root (node 4), there's only a difference of depth 1 between its two subtrees.

The overhead needed to maintain this balance is minimal. A single, 2-bit number is needed in each node and a slight sacrifice is made at insert and delete times. If these times are critical, you may be better off with a normal binary tree. On the other hand, the average-case search time in an AVL tree is guaranteed to be $O(\log_2(N))$, where N is the number of nodes. You usually search a tree more often than you put a node into it, so AVL trees are pretty useful beasts.

Figures 2 and 3, page 22, illustrate how balance is maintained in an AVL tree. There are four possible ways to introduce an imbalance into the tree. Of these, two of them are mirror images of the other two, so they aren't shown in the figures. Figure 2 shows the simplest case. The circles are nodes in a tree, and the rectangles are entire subtrees, which may be made up of any number of nodes. The subtrees will be balanced, however. Figure 2a shows a new node (marked with an X) being added to the subtree labeled alpha, creating an imbalance. The imbalance is corrected in Figure 2b. Note that the subtrees don't have to be modified at all, only the pointers associated with nodes C and E have to be changed. For obvious reasons, this correction is called a (clockwise) right rotation, or RR rotation. The mirror image is an LL rotation.

The harder case is illustrated in Figure 3 (I'll look at the rest of Figure 2 in a moment). Here the imbalance is created by inserting either of the X-

marked nodes into the beta or gamma subtrees. The problem is corrected with a double rotation. First, the tree rooted at B is rotated left (counterclockwise), making the left pointer of node F point at node D. Next, the tree rooted at node F is rotated right (clockwise), bringing node D to the top and moving node F to the right. This final situation is illustrated in Figure 3b. Because I've done a right followed by a left rotation, this is called a double RL rotation. The mirror image is called a double LR rotation.

Figure 4, page 24, shows a series of insertions into an AVL tree. The parenthesized letters tell whether the subtrees are balanced (B), unbalanced to the left (L) or to the right (R).

AVL tree deletion is a marginally harder problem. First, let's look at deletion from a normal binary tree, illustrated in Figure 5, page 26. Node D is being deleted in all three pictures. In 5a and 5b, because there is only one child, the deletion is easy; all you need to do is make the pointer in the parent point around the deleted node to that node's child.

Deleting a node that has two descendants is harder. Notice that the rightmost node of the left subtree (node C) will always have the largest

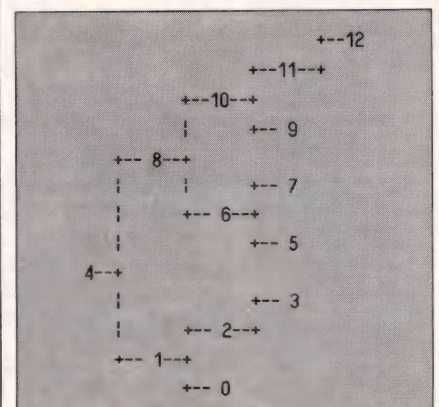
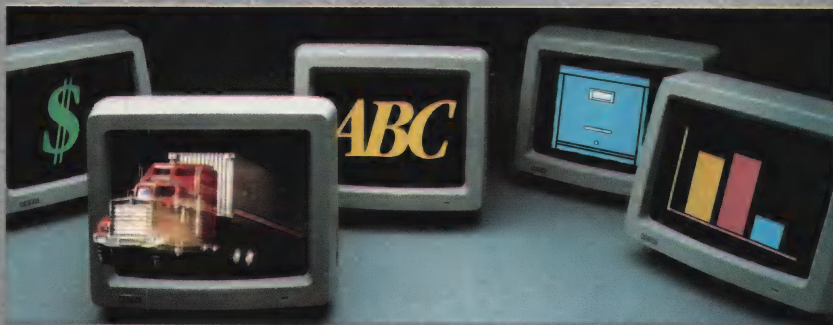


Figure 1: A worst-case balanced AVL tree

Digital has it now.



The first open-ended multi-user software system that integrates your custom application with a full range of standard business programs.

The A-to-Z™ Integrated System for MicroVAX II™ and MicroPDP-11™. Now you can integrate any number of your custom multi-user applications with word processing, graphics, spreadsheet, data management, accounting, and thousands of VAX™ and PDP-11 programs. Same menus, commands and files with no need for a system manager.

Tell me about A-to-Z on MicroVAX™ and MicroPDP-II™ ☐ Reseller ☐ Developer
Name _____ Title _____

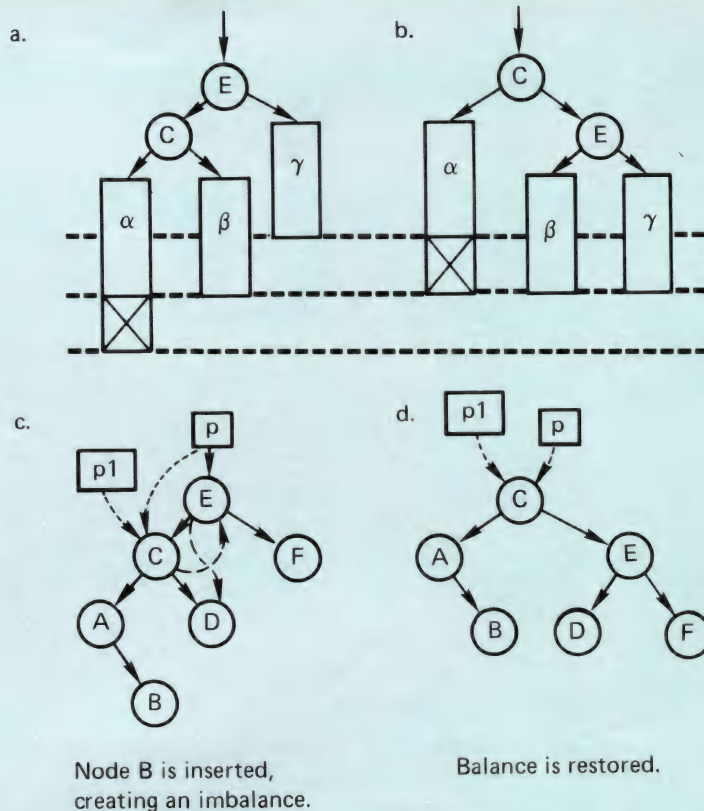
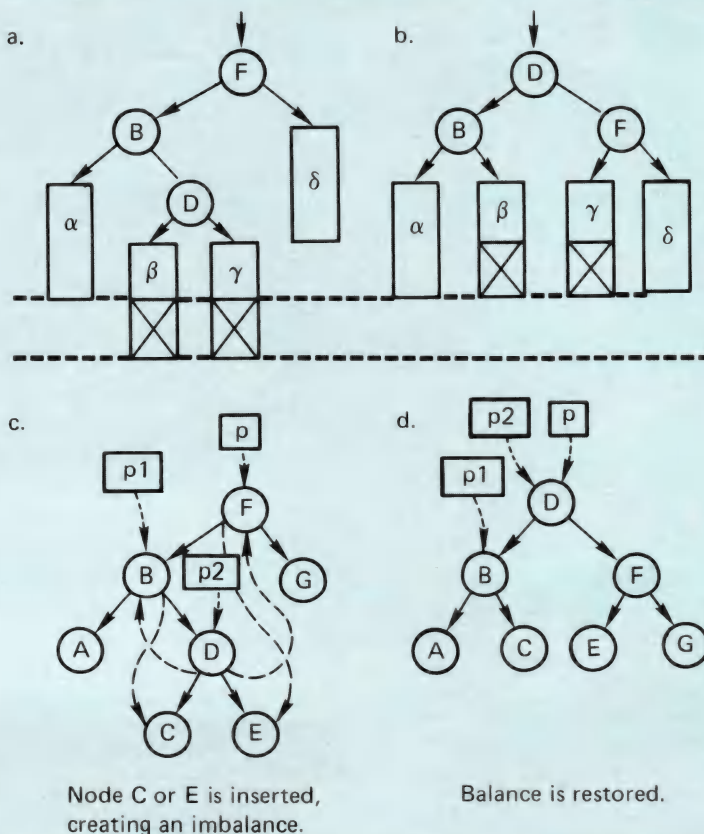
Company _____ Phone _____

Address _____

City _____ State _____ Zip _____

Send to: Digital Equipment Corporation, Inquiry Dept.,
NR02-1/H3, 444 Whitney Street, Northboro, MA 01532.

digital™

**Figure 2: Single rotation****Figure 3: Double LR rotation**

value of any node in that subtree. Also note that all nodes in the right subtree will have values greater than both the node to be deleted (*D*) and the rightmost node of the left subtree (*C*). You can do the delete in one of two ways. The pointers can be modified as shown by the dotted lines in the figure (moving *C* to where *D* used to be). Alternatively, you can transpose the contents of nodes *C* and *D* and then delete node *C* rather than *D*. To do the latter, you descend to node *C* and then copy everything except the pointers from node *C* to node *D*, finally deleting node *C* (I'm actually using this latter approach in the AVL routines).

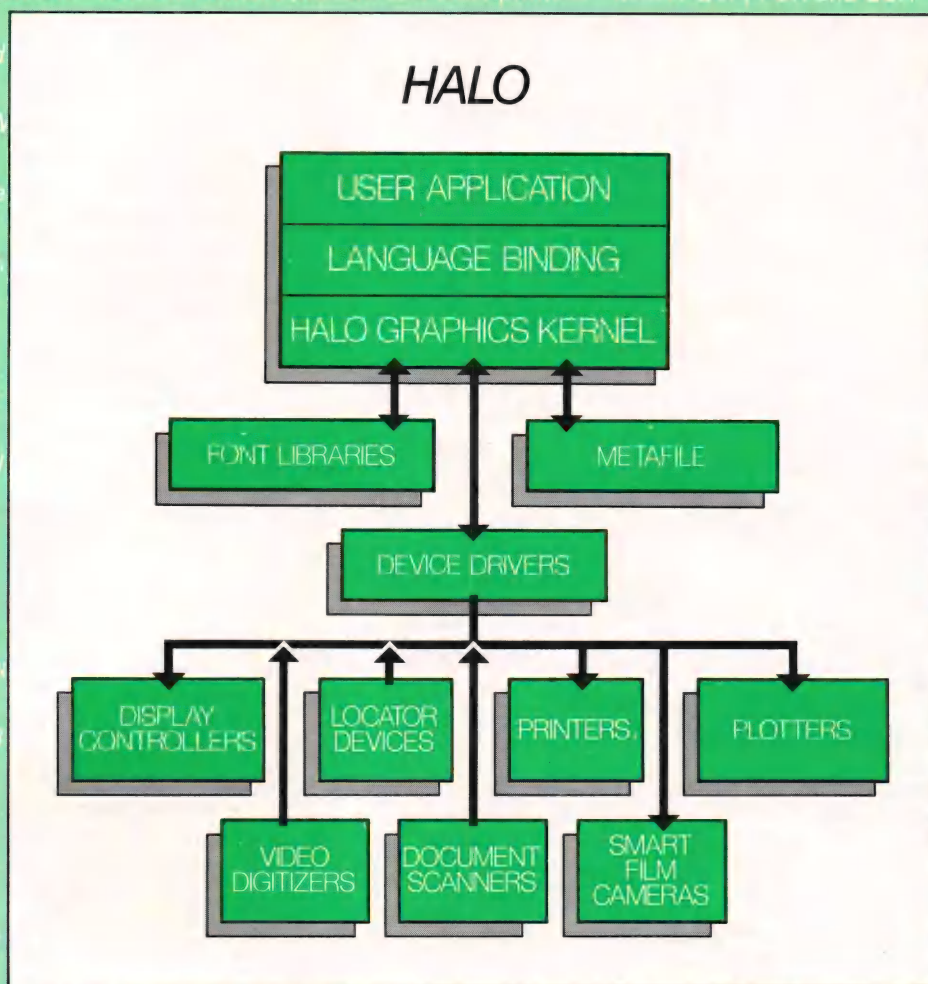
Deletions from AVL trees use the same procedures as are used in a normal tree. As you ascend back up the tree from the rightmost node of the left subtree, however, you must rebalance every node along the way. Fortunately, because you're deleting a leaf, rebalancing is only necessary about one time in five. A series of AVL tree deletions are illustrated in Figure 6, page 27.

Using the AVL Package

Before looking at the AVL routines per se, let's look at an application. I've designed the interface to the AVL routines so that the application program doesn't know that it's using a tree to store data. It interfaces to a series of database routines that store data in unknown ways. The application can insert objects into the database, delete items from the database, find items, and print the entire database, but it has no knowledge about how that database is actually organized. This way if you write a different set of database routines that don't use AVL trees, you don't have to modify the application program (provided that you maintain the same interface to the database routines themselves). Just as the application doesn't know anything about the organization of the database, the AVL tree routines don't know anything about how the application is using that database. All knowledge about the contents of the nodes is passed to the AVL routines as pointers to subroutines.

Tree.h (Listing One, page 86) is a header to be included in all applica-

IN THIS COMPETITIVE WORLD YOU'D BETTER BE USING HALO.



HALO gives you more than lip service to a "graphics standard", it gives you the power and performance today's graphics applications demand.

Freedom Through Compatibility

The HALO graphics subroutine library also gives you the freedom to choose your programming and implementation environment by providing the widest compatibility among programming languages and graphics devices.

Not only does HALO currently support 14 programming languages and 100 device drivers, but HALO keeps you up-to-date with new advances in technology by supporting the latest languages and graphics devices as they become available.

If you want to reach the largest audience and need protection against technological obsolescence, then the HALO Graphics Kernel System is your answer.

History Of Success

The measure of success is acceptance. HALO is supported by the largest installed base of end users, Independent Software Vendors (ISVs), corporate licensees, and Original Equipment Manufacturers (OEMs) of any product of its type.

Full Support

Media Cybernetics offers an 800 number hotline, classes, and on-site training backed up by comprehensive documentation including the acclaimed interactive LearnHalo tutorial.

Ordering Information

- HALO—Single language, single user license, all device drivers, \$300.
- Additional Language bindings at time of order, \$150.
- Font Packs—set of 10 fonts per pack, \$100 each set.

Circle no. 199 on reader service card.

- Special—Order HALO and one Font Pack and get Dr. HALO II for just \$39.95, a savings of \$100.
- OEM ISV and site licensing available, call for details.

To order or request more information, call or write:

Attn: Randy Cambell

media cybernetics, inc.

8484 Georgia Avenue, Suite 200
Silver Spring, MD 20910
(800) 446-HALO, (301) 495-3305



**Graphics tools for
professionals.**

SEE US AT SIGGRAPH BOOTH 1687

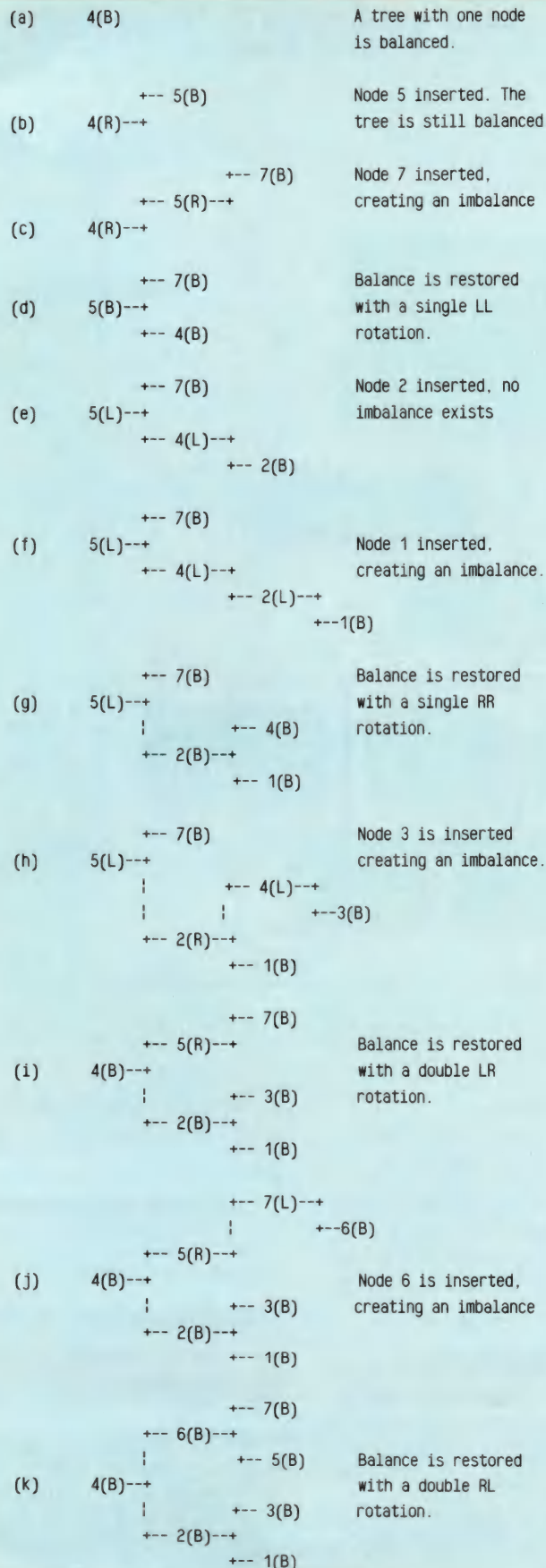


Figure 4: Insertions into an AVL tree

C CHEST

(continued from page 22)

tion programs. It contains nothing but extern statements, useful for function prototyping. The *TREE* type is actually a dummy type—it's used like a *FILE* is used by the standard I/O routines. Your program doesn't need to know anything about the actual organization of either a *FILE* or a *TREE*. The *LEAF* type that's used in the extern statements must be defined in the application program before *tree.h* is *#included* in your program. A *LEAF* is a structure that contains all the fields that your application is interested in. The *LEAF* doesn't need to know anything about trees, so it doesn't need to include pointers to children and so on.

A *LEAF* is defined on line 3 of *test.c* (Listing Two, page 86). It is a structure that contains one field—an integer key. Of course, a more realistic application would have more fields in the structure, but a single key is all that's needed for my example. Skipping ahead, the *main()* subroutine (on lines 84–105) gets a series of commands, first from the command line and then interactively from standard input. It executes these commands by calling *docmd()*. The following commands are recognized:

iN—insert node *N* into the tree
dN—delete node *N* from the tree
fN—find node *N* in the tree
a—delete the entire tree
q—exit back to the operating system

Docmd() (lines 37–80) illustrates how to use all the database routines. *Docmd()* is passed two arguments: *cmd* is a single-letter command that will correspond to one of the cases in the *switch* on line 42; *n* is an integer key that is fetched by *main()*.

The 'i' case (on lines 57–68) illustrates both the creation of a node and the insertion of that node into the database. Memory for the node is fetched from *talloc()* (on line 57). *Talloc()* is used just like *malloc()* is. It returns a pointer to an area of memory of the specified size or NULL if it can't get the memory. In this case I'm allocating a single *LEAF* structure that is initialized on line 61.

The new node is inserted into the database with the *insert()* call on line

QUIT DOING GRUNT WORK.

Let Greenleaf do it for you
and set you free.

C Program developers, stop slaving!

Greenleaf libraries have the functions you need — already perfected and in use by winning program developers in major corporations such as IBM, EDS and GM.

Between our Greenleaf Functions and Greenleaf Comm Library, we have over 340 functions on the shelf. Each one can save you time and effort. Money, too.

Many C programmers have told us that, even if they only use one or two functions, our products easily pay for themselves:

The Greenleaf Functions

The most complete and mature C language function library for the IBM PC, XT, AT and close compatibles. Our version 3.0 includes over 225 functions — DOS, disk, video, color text and graphics, string, time/date, keyboard, new disk status and Ctrl-Break control functions plus many more!

The Greenleaf Comm Library

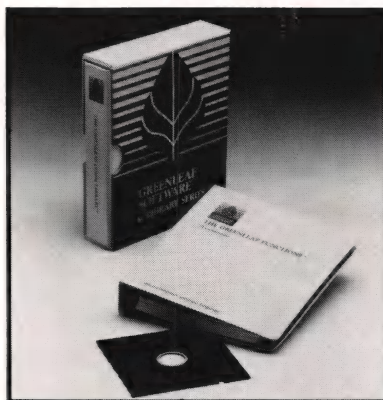
Our 2.0 version is the hottest communications facility of its kind. Over 120 all new functions — ring buffered, interrupt-driven asynchronous communications.

Call Toll Free

1-800-523-9830

In Texas and Alaska, call

214-446-8641



GREENLEAF

Software

**Greenleaf Software, Inc.
1411 LeMay Drive Suite 101
Carrollton, TX 75007**

If you need more than 2 ports, only Greenleaf gives you the total solution — boards, software, and complete instructions that enable you to build a 16-port communication system.

And no matter how many ports you have, it's virtually impossible to lose information with multiple file transfers. XMODEM, XON/XOFF and Hayes modem controls are featured.

We support all popular C compilers for MS DOS: Lattice, Microsoft, Computer Innovations, Wizard, Aztec, DeSmet and Mark Williams.

Order today!

Order a Greenleaf C library now. See your dealer or call 1-800-523-9830. Specify compiler when ordering. Add \$8 for UPS second day air, or \$5 for ground. Texas residents, add sales tax. Mastercard, VISA, P.O., check, COD. In stock, shipped next day.

Greenleaf	
Comm Library v2.0	\$185
Greenleaf Functions v3.0	\$185
Digiboard Comm/4-II	\$315
Digiboard Comm/8-II	\$515

We also sell compilers, books and combination packages.

Circle no. 97 on reader service card.

62. *Insert* is passed three arguments: a pointer to the root (not the contents of the root), a pointer to the node to be inserted, and a pointer to a comparison function. This function is defined on lines 23–28. It's passed two pointers to *LEAF*s, and it works like *strcmp*() does, returning a negative number if the left argument is less than the right argument, 0 if the arguments are equal, and a positive number if the left argument is great-

er. In the case of two structures, the relative value is determined by comparing the key fields.

Insert() returns NULL on success, and it returns a pointer to a conflicting node if a node having the indicated key already exists in the database. In this application, the new node is just discarded (with the *tfree*() call on line 65) if a conflicting node exists. In a more sophisticated application, you might want to update a count field in the conflicting node or do some other operation.

The 'd' case (on lines 44–47) deletes

the node having the indicated key from the database. *Delete*() returns 0 if the requested node isn't in the tree. *Delete*() is passed three arguments: a pointer to the root, the key to be deleted, and a pointer to a comparison function (*dcmp*). This comparison function is defined on lines 29–33. It is passed *delete*()'s second argument (that's all that *delete*() does with its second argument—passes it to the comparison function) and a pointer to one node in the tree. It compares the two arguments and returns a negative number if the key is less than the node, 0 if the key equals the node, and a positive number if the key is greater than the node (just like *strcmp*() does). I could have used the same comparison function used by *insert*() here, but I would have had to create a dummy *LEAF* just to put the key in it and then pass a pointer to that dummy node to *delete*().

The call to *freeall*() (on line 71) deletes the entire database, as compared to a single node in the database.

The 'f' case does a find function. *Find*() returns a pointer to a *LEAF* if a structure having the indicated key is in the database; otherwise it returns NULL. *Find*() is passed the contents of the root (unlike *insert* and *delete*, which are passed pointers to the root); its other two arguments are the same as *delete*()'s.

The remaining subroutine of interest is *tprint*() (on line 78). *Tprint*() prints out the entire tree graphically, using a slightly modified version of the graphic *inorder* traversal routine I looked at last month. That is, it prints the trees as they are shown in Figures 4 and 6, with lines showing all the pointer relationships between nodes. *Tprint*() differs from last month's routine in two ways. First, it's passed an output stream. If that stream is *stdout*, and *stdout* is not redirected to a file, then IBM box-drawing graphic characters are used instead of plus signs and dashes. Next, it is passed a pointer to a print subroutine (because the database routines don't know anything about the contents of a *LEAF*, you have to pass it a print routine). This routine (defined on lines 9–19) is passed an output stream and a pointer to a *LEAF*. It should print the node's contents if that pointer is non-NULL; otherwise it should print as many space charac-

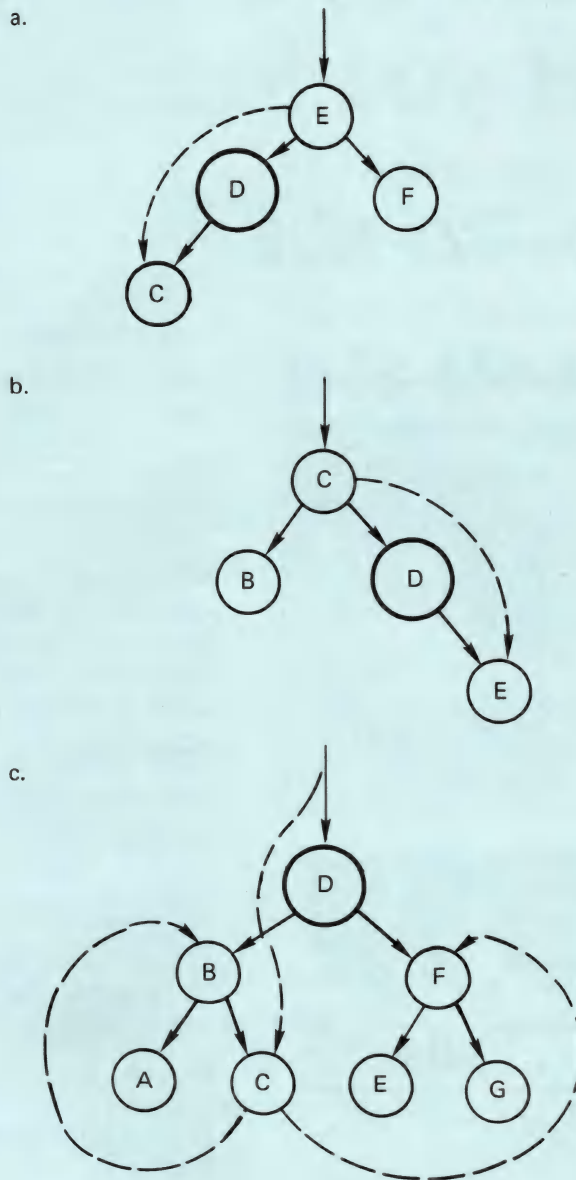


Figure 5: Tree deletion

ters as it would use to print these contents. That is, it must always print the same number of characters. Sometimes those characters are the node's contents, and sometimes they're spaces, depending on whether the *LEAF* pointer is *NULL*.

The AVL Routines

The AVL tree functions are in Listings Three through Eight, pages 87–102. I've merged all these routines into a single library and linked that library with *test.obj*. Listing Nine, page 102, is a makefile for creating both library and *test.exe*. I'm using the Microsoft C compiler and Polymake, but most of the commercial *makes* will work.

Avl.h (Listing Three, page 87) is a header file used by all the library routines but not by the application program. A tree node is defined by the *HEADER* structure on lines 1–8 and is pictured in Figure 7, page 28. *Talloc()* (Listing Seven, page 90, lines 20–34) uses the same sort of data structure that *malloc()* uses to keep track of the free list. It gets the amount of memory that the user needs plus enough memory for a *HEADER*. So the data space is extra memory concatenated onto the end of a *HEADER*. As you can see in Figure 7, *malloc()* returns a pointer to just below the header that it uses. Similarly, *talloc()* adds its own header to this memory and returns a pointer to just below that second header. *Tfree()* (on lines 38–42) is passed the pointer that *talloc()* returns. It decrements that pointer to get back the pointer that *malloc()* returned.

In the small model, a *talloc()* *HEADER* uses 6 bytes (two pointers and an *int*). The *malloc()* header (if Microsoft's *malloc()* is anything like the one in K & R) uses 4 bytes (one pointer and one *int*). So a total of 10 bytes (in addition to the amount of memory needed for the data area) are required as overhead in each tree node. Note that, as is the case in *malloc()*, the application program doesn't know (or care) anything about the contents of the *HEADER*, so the database routines can put anything they like up there without affecting the application program.

A *HEADER* contains four fields: *left* and *right* are pointers to the subtrees; *size* is the size of the application area (the parameter passed to *talloc()*)—it's

used by the *delete* routine). The *bal* field is used to keep track of the relative sizes of a node's two subtrees. In most AVL routines the balance factor is determined by subtracting the height of the left subtree from the height of the right subtree. Because of the way that AVL trees work, this number will always be -1 , 0 , or 1 . In order to be able to put the balance factor into a bit field, which must be an *unsigned int*, I've added 1 to this equation, shifting the three numbers up to 0 , 1 , and 2 . *Bal* is 0 if the left

subtree is larger, 1 if the subtrees are the same height, and 2 if the right subtree is larger. To make the code more readable, these three numbers are *#defined* as *L*, *B*, and *R* (left, balanced, and right) on lines 12–14 of *avl.h*. The remainder of the *include* file is extern statements for all non-static (globally accessible) subroutines in the various tree files.

The tree-printing routine (*tprint()*) is in Listing Four (page 87). One of two character sets is used to print the connecting lines. *Graph_chars* (line 38)

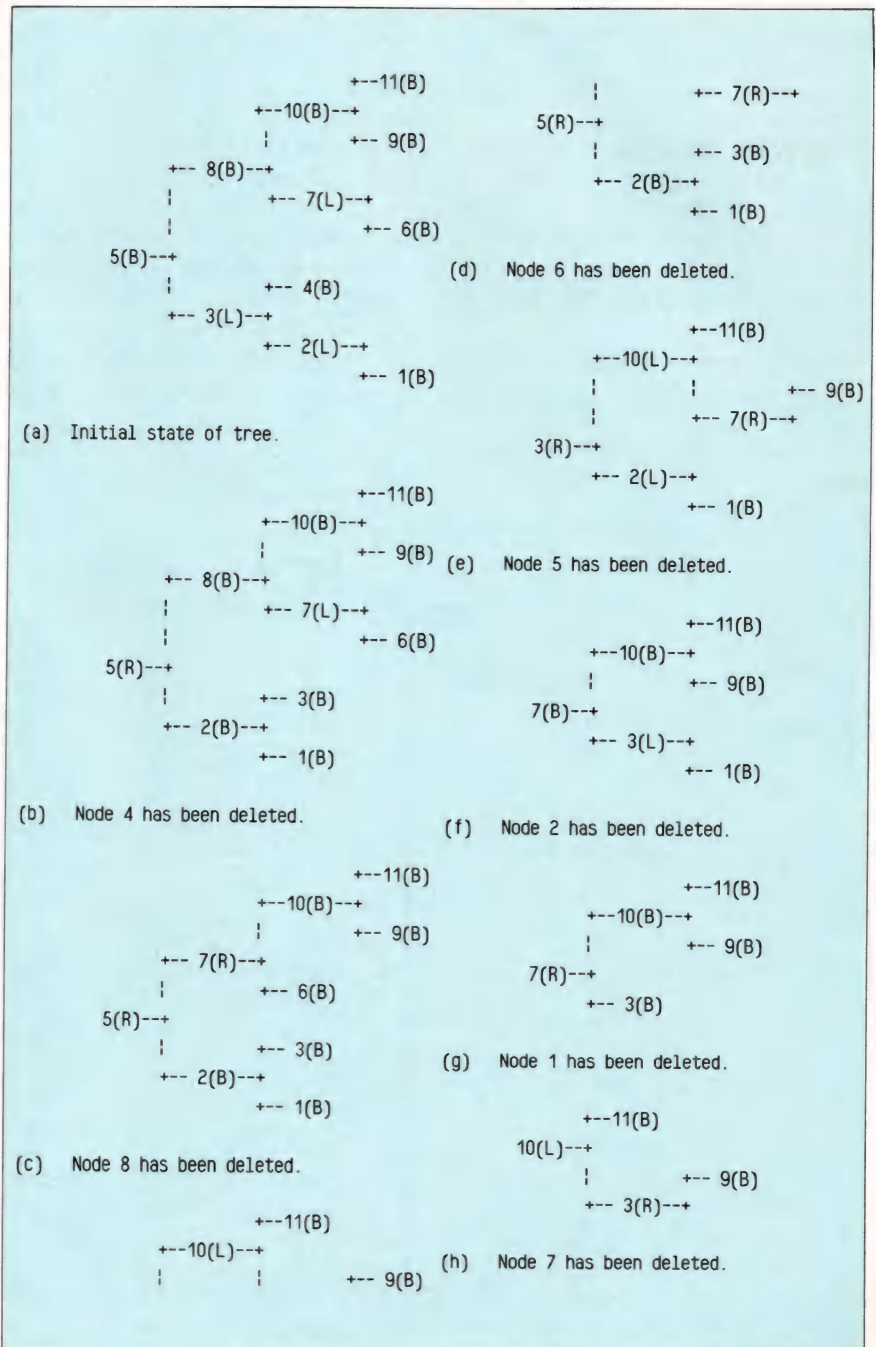


Figure 6: AVL tree deletion

contains the IBM box-drawing graphic characters. *Norm_chars* (line 41) contains plus signs and dashes. *Cset* (line 45) will point to one or the other of these sets, and the macros on lines 21–26 are used to get at a specific string. There's one for each of the various corners and tees that are needed to draw the tree.

The macros on lines 28–34 are debugging diagnostics. Defining them in this way makes the code cleaner as *PAD* and *PBAL* will expand to empty strings if *DEBUG* isn't *#defined*. This way you don't have to litter the code with *#ifdef DEBUG* statements.

The general-purpose, bit-map routines used last month have been replaced by the *testbit()* macro and the *setbit()* subroutine on lines 52–61. They work in the same way as last month's routines do, but they're smaller.

The traversal routines are recur-

sive, and as is generally the case with recursive routines, I've tried to minimize the size of the stack frame by putting as many variables as possible into global statics. *Tprint()* (lines 125–135) is an access routine, initializing these global variables and then calling the workhorse function *trav()* (lines 66–121) to do the work. *Tprint()* decides which character set to use on lines 132–133. If the output stream is *stdout* and this stream is attached to a device (as compared to a file), graphic characters are used. So, graphic characters will be used if *stdout* is not redirected to a file. Note that *isatty()* returns true if *stdout* is assigned to any device, so graphics will be used if *stdout* is redirected to the printer. I've used *isatty()* because it's easy. If you prefer you can use DOS function *0x44* (*IOCTL*) to see if the device is actually the console or not (as compared to a printer or whatever).

Trav() uses the same algorithm as *inorder()* used last month; consult last month's column for details of its

operation. The only differences have to do with getting either the graphics or normal characters printed at the appropriate places (and the printing is done indirectly through a pointer to a print subroutine).

Find() (Listing Five, page 89) does a simple, recursive descent into the tree. From *find's* point of view, it doesn't matter that the tree is an AVL tree because it doesn't need to use either the *size* or *bal* fields of the *HEADER* structure. Similarly, *freeall()* (Listing Six, page 89) also does a recursive postorder traversal, freeing the memory used by the current node after freeing both the right and left subtrees.

The hard-to-do functions are inserts and deletes. *Insert()* is in Listing Seven, lines 148–166. As was the case in *tprint()*, *insert()* is actually an access function that initializes a few static global variables and then calls the static workhorse function *ins()*, on lines 46–144. The algorithm used here (and in the *delete* function below) are straightforward translations of the Modula-2 code in *Algorithms & Data Structures* (Niklaus Wirth, Englewood Cliffs, N.J.: Prentice-Hall, 1986) pp. 218–227. I've cleaned up Wirth's code considerably, but the algorithms are the same. (It's a mystery to me how the inventor of Modula-2 can write such miserable code in Modula-2.)

The static variable *h* (declared on line 56) is a little weird. It will change its value magically after every recursive *ins()* call to reflect whether the tree has grown in depth as the result of an insert. In this case a rebalance of some sort is needed. Left subtree rebalancing is done on lines 80–103 and right rebalancing on lines 110–140. Figures 2c and 2d illustrate how the pointers are juggled to do an RR rotation. Figures 3c and 3d illustrate a double LR rotation. The rebalances are done as you ascend back up the tree from the newly inserted node.

The *delete()* function is in Listing Eight (page 96). The access function is at the end of the file (lines 220–240); the workhorse function (*del()*) is on lines 165–216. *Del()* descends to the node to be deleted in the usual way on lines 182–192. *Got_smaller* is used in the same way as *h* was used in the *insert* function. Here, though, you're noting whether the tree has shrunk

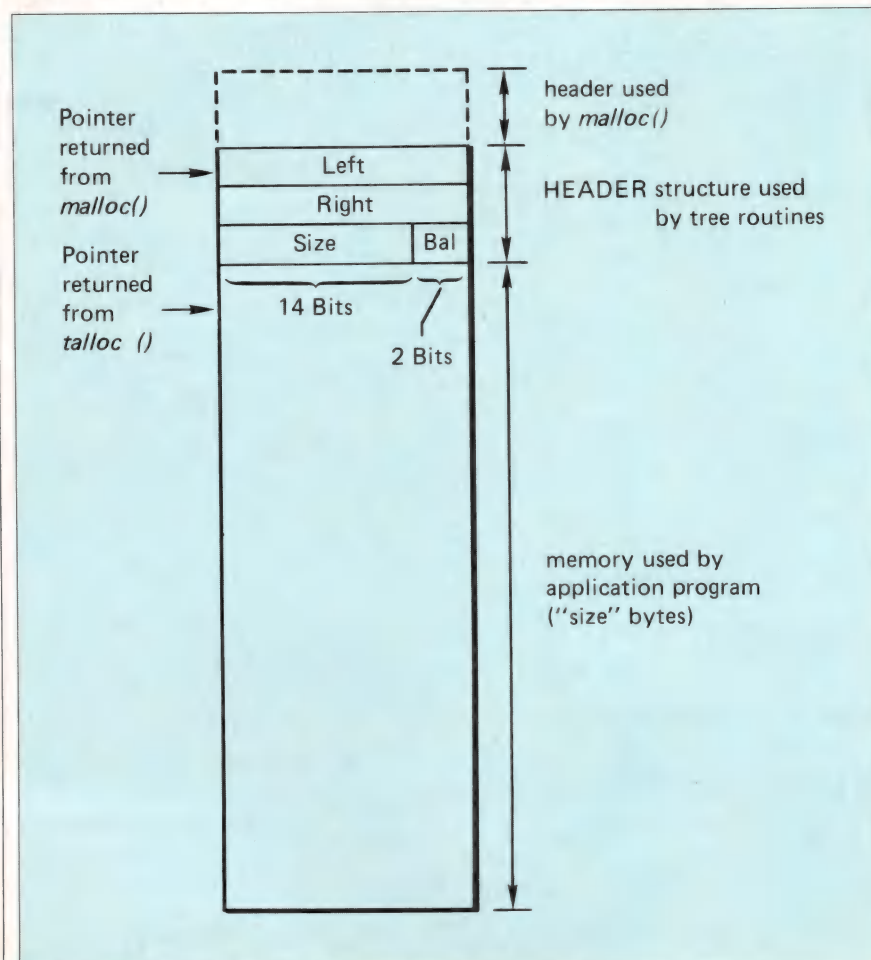


Figure 7: Tree node

rather than grown. The calls to *balance_l()* and *balance_r()* (sources on lines 21–135) do a left or right rebalance when appropriate. These routines use pretty much the same code that was used in the *insert* function. You might want to make *insert()* call these routines rather than using the imbedded code if space is tight.

Once found, the node is actually deleted on lines 194–212. *Dp* points at the node to be deleted. The two easy cases (pictured in Figures 5a and 5b) are done on lines 196–204. The hard case, in which the current node has two children, is done by *descend()*, called on line 206.

Descend() (lines 139–161) descends recursively to the rightmost node of the left subtree (it's passed a pointer to the left subtree the first time it's called). When it gets there, *descend()* copies the contents of that node up into the node to be deleted (pointed to by *dpp*) and modifies *dpp* to point at the current, rightmost node (on line 157). It then ascends the way it just came, rebalancing as it goes up.

Conclusion

So that's AVL trees. The code is obviously more complicated than that for simple binary-tree routines, but the complexity is worth it if you need fast access time into the tree. Next month I'll round off my discussion of trees by looking at a directory-tree printing routine that graphically prints a map of your hard disk's directory tree.

Availability

All the code published this month is available both on CompuServe (type go ddjforum) and, for \$25, on an IBM PC-compatible disk from Software Engineering Consultants, P.O. Box 5679, Berkeley, CA 94705. The tree routines from last month are on the same disk.

DDJ

(Listings begin on page 86.)

Vote for your favorite feature/article.
Circle Reader Service No. 2.

Order our FULL C COMPILER For \$59⁹⁵ and we'll give you a free CED Program Editor



The Ecosoft Eco-C88 compiler for the 8088 and MSDOS is going to set a new standard for price and performance. Consider the evidence:

Compiler	Eco-C88	Lattice (1)	C86 (1)
Seive	12	11	13
Fib	43	58	46
Deref	14	13	—
Matrix	22	29	27
Price	\$59.95	\$500.00	\$395.00

(1) *Computer Language*, Feb., 1985, pp. 73-102. Reprinted by permission.

Eco-C88 Rel. 3.0 on IBM PC with 2 floppy disks, 256K. Benchmarks from Feb., 1985, *Computer Language*.

Eco-C88 includes:

- ★ All operators and data types (except bit fields)
- ★ Prototyping, structure passing and assignment, enum and void language enhancements.
- ★ Tiered error messages (gives you selectable levels of "lint" semantic checking)
- ★ memfiles (TM) for using memory outside the 128K limit as a file
- ★ Expanded library with over 200 functions (many of which are System V compatible) plus color and transcendental
- ★ ASM or OBJ output; uses the MSDOS linker
- ★ 8087 support with 8087 sensed at runtime
- ★ cc and "mini-make" for easy compiles (with source code)
- ★ expanded user's manual

If ordered with the compiler, the C library source code (excluding transcendental) is \$25.00 and the ISAM file handler (as published in the *C Programmer's Library*, Que Corp.) in OBJ format is an additional \$15.00. Please add \$4.00 for shipping and handling. To order, call or write:



Ecosoft, Inc.
6413 N. College Avenue
Indianapolis, IN 46220

(317) 255-6476 • 8:30-4:30
1-800-952-0472
(orders only)



Circle no. 89 on reader service card.

IBM PC/XT/AT SYMBOLIC DEBUGGER

"C" FOR YOURSELF!

ONLY **CodeSmith-86** has **AutoBrowse®**

ORDER (800) 732-2345 NOW

The *effortless* way to debug your **C, PASCAL, & FORTRAN.**

Step & Break thru synced **SOURCE and Machine CODE WINDOWS.**

☒ Dual-mode **PATCHING ASSEMBLER**

☒ 4 STOP-on-Data-Compare conditions

• 256K RAM min.

• DOS 2.0 → 3.x

CodeSmith-86 the original Multi-Window Debugger

GUARANTEED 30 DAYS OR YOUR BUCKS BACK

"CodeSmith really shines..." —PC TECH JOURNAL

"It has greatly speeded up the development of new programs and the maintenance of existing programs." —MicroPro Int'l Corp.

"...has saved us uncountable hours in debugging communications software." —Dave Adkins, Jim Dukat, Los Altos CA

"...has almost all the features that are in the Hewlett-Packard 64000 emulator I use at work." —Joel Lagerquist, Riverside CA

"CodeSmith-86...a superb utility." —Steven Kang, New York NY

VISUAL AGE ♦♦ In Calif. (213) 534-4202

642 N. Larchmont Blvd. ★ Los Angeles, CA 90004

\$145.00

Circle no. 291 on reader service card.



BENCHMARKING C COMPILERS

by Richard Relph, Steve Hahn, and Fred Viles

This review is based on the best information available as of the article deadline, May 15. Running Light, page 8, which has a later deadline, contains updated information received by June 1. The two months between our deadline and our August publication date can be a long time in the competitive C compiler market, in which an adequate product may become superlative in the next version. We will be publishing updates as often as is necessary and will be maintaining even more information on the compilers in the DDJ Electronic Edition on CompuServe. See DDJ On Line, page 18, for more information about the Electronic Edition. The Editorial, page 6, contains a list of the vendors' addresses and phone numbers.—eds.

***Documentation has improved
across the board, as has
compiler quality.***

We would like to thank especially those who worked Saturdays and evenings to review documentation and package compilers. These people are Joe

Marshall, Peter Vutz, Scott Thomas, Phil Freiden, Stan Peters, Fletcher Johnson, and Don Hackler. Other C-SIG members also contributed.

Last year some of us participated in the writing of a similar review (DDJ, August 1985). After the review was completed, we approached Manx Software Systems in order to arrange a group buy of its package, one of the best last year. As a result, nearly all of this year's reviewers own Manx Aztec C. To counter this, and the other reviewers' personal ownership of any given C compiler, reviewers were prohibited from making any subjective comments regarding compilers they owned. We feel that, by making these statements up front, we will put your mind at ease regarding our objectivity.

Last, the principal authors of this review are professional programmers. We make our living writing C programs. Consequently, this review is geared toward professional programmers.

The Compilers

Since our last comparative review of C compilers, three vendors have left the marketplace, five vendors have products to be reviewed here for the first time, and all others have new versions or products. In general, things are not the same as last year, and some results are surprising. Documentation has improved across the board, as has compiler quality. Libraries are bigger and better than last year.

Microsoft, Lattice, Mark Williams Co., Ecosoft, Datelight, and Wizard Systems Software provided substantially changed (major version number) compilers for review.

This is a review of 17 C compilers for MS-DOS from 15 different companies. Its intent is to provide knowledgeable C programmers with insights into each of the products otherwise unobtainable without purchasing the compiler. It is a very detailed study, and more casual C programmers may find some of the details confusing or of little value. We have therefore attempted to put the generally interesting information at the beginning of each major section, just prior to a gradual (or rapid) descent into the nitty-gritty.

The Reviewers

Any review of this magnitude is clearly a difficult task. Were it not for the assistance of many people, including the vendors themselves, it would have been impossible.

Richard Relph, 8345 Springdale Ct., Gilroy, CA 95020; Steve Hahn, 63 Sheridan Rd., Oakland, CA 94618; Fred Viles, 1854 Half Pence Way, San Jose, CA 95132.



Manx and C Ware Corp. both added significant features to their compilers with only minor version number changes. Software Toolworks has added *long* and *float* support as an add-on option, which we reviewed. Only Computer Innovations has approximately the same package as last year. C-systems, Control-C Software, and Digital Research have stopped marketing MS-DOS C compilers. New to the review this year are Whitesmiths, MetaWare, WordTech Systems, Mix Software, and IBM. Companies queried about having their compilers reviewed, but declining, were Supersoft and Codeworks. Telecon Systems could not be located.

All in all, 15 companies provided compilers—two of them (Mark Williams and Datalight) provided two, bringing the number of packages reviewed to 17. Some vendors opted to send beta versions of compilers due to be out by the time you read this. Beta testing gives a vendor a chance to test a product more thoroughly before release to the buying public. We found bugs in all beta compilers and reported them, both here and to the vendor. Unless otherwise noted, all bugs were expected to be fixed before final release.

The Benchmarks

Results of benchmark tests of compilers have a justifiably bad reputation. All too often performance figures are cited out of context or overgeneralized into overall ratings; all too often the tests are misapplied, incorrectly performed, or inadequately documented. We have attempted to provide repeatable measures of performance of the compilers on tests chosen to detail the strengths and weaknesses of the individual compilers.

We would have liked to run all the benchmarks on all available machines, but this proved to be impossible. Instead, we ran them on an AT&T 6300 with 640K and an 8087. All benchmarks were run on a RAM disk, with careful attention paid to the layout of the storage. The AT&T has an 8-MHz clock and a 16-bit bus. The processor was not the customary 8086 but a NEC V30. The effect of using

the V30 is discussed later.

This year, we did manage to test all the compilers for conformance to some standard. To do so required the use of a validation suite, and we were fortunate enough not to have to write one. The source of the suite is also a reviewee (MetaWare), however, and this should be noted. We polled some other compiler vendors to see if they had any objection to our using a competitor's test suite, and the only point made was that failing a given test should not be treated as significant unless acknowledged so by either K & R or the failing compiler's vendor.

All our execution-time benchmarks have the same basic structure. Each test function starts by initializing whatever it wants to, starting the timer, doing the task, getting the current time (relative to start), doing any wrap-up chores, and returning the time. To start the timer, we have provided each compiler with a *time_0* function and a companion function, *time_n*, which returns the time elapsed since the last *time_0* call in tenths of seconds. The number of times that the task is done is controlled by a loop count passed into the test function from main.

Main is made independent of the test to be run by having it reference an array of structures, *bm*, which each source file that contains an execution benchmark contains. The structure has a pointer to the function to be tested, the name of the function (for reporting purposes), the loop count, and a place to keep the return value (the execution time). Main steps through this array, calling each function in turn, passing it its loop count, and saving the return value in the structure. After each function call, main dumps the results for that function to the screen. When it reaches the end of the array, main opens a results file and writes the results out so that we can manipulate them later. There is a special version of main for the 8087 tests, which multiplies each loop count by 10 before passing it to the test function.

When reviewing the results from these benchmarks, be sure not to add up either total wins or total execution time for the whole set. To do so would assume that the



C COMPILERS

(continued from page 31)

sum of the contrived tests we have thrown together represents a "typical" program, which it certainly does not. Instead, go through the descriptions of each of the benchmarks and decide how much each feature tested is present in your code.

The closest thing to a typical program in this benchmark suite is the dhrystone, but even its results (see Table 4, page 41) must be taken with a large block of salt. The dhrystone as presented here is about 20 percent dependent on how string library functions are implemented. MetaWare's compiler, for example, turns in 892 dhrystones per second when calling its very fast (based on the results of string) string library functions. But when dhrystone is recompiled with MetaWare's string.h file, which maps the string functions onto special compiler-recognized functions for in-line code generation, its compiler gains 184 dhrystones per second, giving 1,076. Wizard's compiler is near the bottom of the list of dhrystone contenders but has excellent results in the other compiler benchmarks. The reason is that, when we ran our tests, Wizard had written its string package in C. When it changed this to use assembly language, this compiler, too, added about 20 percent to its dhrystone score. This apparent flaw in the benchmark (when compared to the comments, which claim no library dependency at all) is because dhrystone was originally written in Ada,

which allows string assignment and comparison in the basic language.

Here are brief descriptions of the benchmarks we used in this review:

- array tests the compiler's ability to access arrays using conventional array operations. In it we copy one $10 \times 10 \times 10$ -integer array to another using three nested *for* loops.
- atox tests three functions—*atoi*, *atol*, and *atof*—in a single benchmark. It has 21 *atoi* calls, 16 *atol* calls, and 8 *atof* calls. Each call passes a string constant, some of which have large numbers of lead 0s or blanks.
- cpyblk copies one 10,000-byte file to another in 1,024-byte blocks using *fread* and *fwrite* in a simple *while* loop.
- cpychr does the same thing but uses *fgetc* and *fputc*. Comparing the times will tell you whether block or character I/O is faster for a given compiler.
- diskio tests the speed of *fseek* operations inside a 240K file. The test is based on a benchmark published by Houston, Broderick, and Kent in the August 1983 issue of *Byte*. We've changed it so that the file is created externally with a pattern in it so that data read and written at each position can be verified.
- doc1 through doc1000 are used to measure compiler speed. Doc1 gives you a base for the compiler load and start-up time. It consists of the single declaration *int x*;. Doc1000 is the other end—a 1,000-line program. By subtracting doc1 compile time from that for doc1000, you obtain the time to compile 999 lines of source.
- fibtest is the standard recursive Fibonacci number generator. We pass 24 into it. A good result here means a compiler generates good function entry and exit code. Note that fib does not have any locals and has only one argument.
- fillscr tests the speed of screen output in the absence of scrolling. It uses *puts* to write 1,248 characters per loop to the screen, gradually filling it.
- We have four functions to test function-calling overhead. The functions call another function with zero, one, two, or three parameters. The called functions do nothing but return. Each loop does 500 function calls, and we did 10,000 loops for each function.
- To test function-return capabilities,

we created a function *dfuncrct*, which returns a *double*. The value returned is 0. The call looks like *ret = xfunc(ret);*. There are 1,000 calls to *xfunc* per loop. *Dfuncrct* is run with 250 loops in software floating-point environments or with 2,500 loops in 8087 environments.

- looptst is a simple test of *for* loops. Each loop does an inner loop 10,000 times. We ran this test with an outer loop count of 500. The purpose of *looptst* is to find out about a compiler's loop overhead.

- memory is our test of the *malloc*/*free* function pair. For each of 500 outer loops, we *malloc* 500 50-byte spaces, *free* every fifth one, then *malloc* 100 35-byte spaces and *free* all the blocks.

- The min series are tests to measure library granularity. We have a *minmain* to provide a base line number for start-up code and termination code. We add calls to *printf* in *minprtf* to see what sort of space penalty must be paid for this frequently used function. Where easy, we exclude floating-point routines. *Minputs* replaces the *printf* with a simpler *puts* call. To see the effect of more explicit calls to *fopen*, *fgetc*, *fputc*, *fread*, *fwrite*, and *fclose*, we use *minfio*.

- optimize is our weakest benchmark. It is used to measure the differences between dumb and smart compilers. Although a good time here is positive, the benchmark is extremely artificial and is prone to being reduced to virtually nothing by relatively simple optimization techniques. We did make each compiler provide assembly-language output for this test so we could compare the various optimizers. In the future, the whole area of optimization should be the subject of a separate test suite in which each function will test for the presence of a specific optimization.

- pointer is an attempt to duplicate arrays with pointers without getting rid of the three levels of indirection. What we ended up with is not very representative, but it does test pointer dereferencing fairly well. We have the same two $10 \times 10 \times 10$ -integer arrays from the array benchmark. We add six pointers (three each for the source and destination arrays) and set up the first pointer to actually move through the array. A

FOR C
AND UNIX

WINDOWS FOR DATA™

PC DOS/UNIX
COMPATIBILITY

MENUS WINDOWS DATA ENTRY

PROFESSIONAL DEVELOPERS, here is the front end package you've been waiting for, the ONE that does the hard jobs that others can't — we **guarantee** it. Makes standard display and entry tasks easy. Reliable. Compact. Portable.

MENUS: Build multi-level menus in the format of your choosing: Lotus 1-2-3, Macintosh, or any style you might select. Items can call sub-menus, data-entry windows, or action functions. The menu system is completely flexible.

WINDOWS: WFD is built upon and includes **Windows for C**, the windowing system rated #1 in PC Tech Journal (William Hunt, July 1985). WFC now has more features than ever: unlimited windows and files, pop-ups, fast screen changes, window names, horizontal and vertical scrolling, logical video attributes, highlighting, support for the EGA, off-screen updating, formatted output, word-wrap and auto scroll, print windows, read and write functions, and keyboard input with subroutine execution during waits.

DATA ENTRY: The most complete and flexible data entry system on the market. Pop-up data-entry windows, field types for all C data types, plus special decimal, date, and time fields, full-featured field editing, auto conversion to and from strings for all field types, input masks or "pictures," protected text, system and user-supplied validation functions, range-checking, scrollable context-sensitive help, required and must-fill fields, passwords, programmer definable keys, field types, and mask functions, date and time utilities, and string utilities. Read field by field or auto-read all fields. Branch and nest window forms.

FLEXIBLE



As many possibilities as Vermont in June.

WINDOWS FOR DATA HAS UNPRECEDENTED FLEXIBILITY. Virtually every capability and feature of WFD can be modified to meet special needs. All key-invoked data-entry functions can be assigned to keys of your choosing; and you can add your own functions to the

key assignment table. This same flexibility exists for the input masks used to control data entry. Install your own validation functions. You can even define new field types and add them to the system. You do not need source code to take advantage of the flexibility of WFD, but **full source is available.**

MICROSOFT WINDOWS COMPATIBILITY

is automatic. **Windows for Data** detects the presence of MS Windows (and IBM's TopView) and follows the rules required for full compatibility. No need for special code, complex interfaces, or expensive toolkits.

EASE OF USE

Ease of use comes first and foremost from basic design and implementation. WFD is not just a library of functions, but an integrated system for menus, windows, and data entry.

We make the system easy to learn by explaining each major application in step-by-step detail. WFD is documented for the professional. Six hundred pages of documentation in a full-size, high-quality binder. Numerous tutorials and demonstration programs are provided. Nearly two hundred functions are documented individually, to UNIX standards.

RELIABLE



As free from bugs as Vermont in January.

For its basic input and output, WFD uses the library of **Windows for C**, a mature product that has earned a reputation for extreme reliability. WFD has been through its "shakedown," and the few bugs that turned up have been corrected. We promise to quickly respond to any further bug problems you may encounter.

THE MEMORY FILE FACILITY

of windows for C is more flexible and memory efficient than the "virtual screen" systems of other windowing packages. Memory usage adjusts to the amount of text

in files. **No waste space!** Build files of any length and width from disk, code, or communications input. Retrieve, replace, add, and scroll file lines. Open windows at any point in a memory file. Scroll windows horizontally or vertically.

UNIX, DOS, OR BOTH

WFC and WFD provide source code compatibility between PC DOS and UNIX. Programs written for one operating system will compile and run on the other with only minor changes.

UNIX developers, now you can put advanced windows, menus, and data entry features common to the PC world in your UNIX programs.

PRAISE FROM USERS

"WFD is the best programming tool I've ever used. It's the most flexible I've seen. Whenever I've wanted to do something, I've been able to find a way."

Steven Weiss, Stratford Systems (18 yrs; 1 yr)*

"WFC is the standard by which we judge all other C utilities. The most helpful tool we've ever acquired. Absolutely easy to use. Very tight code." James Baker, Mathew Bender (7 yrs; 4 yrs)*

"Especially compared to **Panel**, I love **Windows for Data**. Your documentation is great." Don Heinmeller, Law Software (10 yrs; 4 mo)*

"The documentation lets you get up and running fast. I integrated help routines into existing educational programs in a day and a half."

Richard Rovinelli, Educational Services (17 yrs; 1 yr)

* (programming experience; C experience)

If you are tired of screen utilities that are hard to use and limited in capability, you owe it to yourself and your programs to try **WINDOWS FOR DATA**.

OUR CHALLENGE AND GUARANTEE

If you have an application where no other tool can do the job, try **Windows for Data**. If it doesn't help you solve your problem, RETURN FOR A FULL REFUND. YOU MUST BE SATISFIED.

WINDOWS FOR DATA WINDOWS FOR C

PC DOS*	\$295	\$195
XENIX-286	\$595	\$395
UNIX	CALL	CALL

* For all popular C compilers;
No royalties for DOS



Vermont Creative Software
21 Elm Ave.
Richford, VT 05476
802-848-7738
ext. 31

MasterCard & Visa Accepted. Shipping \$3.50
VT residents add 4% tax.

Trademarks: Panel, Roundhill Computer Systems; Microsoft, (registered) Microsoft Corp., TopView, IBM.

C COMPILERS

(continued from page 32)

second pointer points at this, and the third points at the second. Then the body of the inner loop looks like `***dp3 = ***sp3`, and the `for` statement moves `sp1` and `dp1` through the arrays.

•`prtf` is an attempt to measure `printf` conversion efficiency, but it is mostly bound by console I/O limitations. Because of this, the `prtf` results should be compared to the `scroll` results to

find the overhead directly attributable to `printf` conversions. The line lengths are the same in `prtf` and `scroll`.

•We think the sieve is a useful benchmark, just not one that any reasonable C programmer would have written without register variables. Therefore, we provide two versions—the standard sieve, without register variables, and our `rsieve`, which has register variables `i` and `k`. We do 140 iterations rather than the normal 10 so that our timer will have

something fairly significant to measure.

•`scroll` is used to see if any compilers differ in the way screen scrolling affects them. It is identical to the earlier `fillscr` except that a new line replaces the bare carriage return done every 78 characters.

•We have a series of tests to see how the compilers differ in treatment of various storage classes. We have a test with four automatic variables, a test where the variables are declared *static*, a test where all are declared *register*, and a test where only two are declared *register*. The purpose of the last test is to see if any compilers implement more than two registers. If any do, then `regtest` will be different from `reg2test`.

•To test some basic string manipulations, we use our strings benchmark. For each loop, we call `strcat` four times; `strcpy` twice; and `strncpy`, `strlen`, `strcmp`, and `strncmp` once.

•`tdouble` and `tfloat` are used to measure basic floating-point speed. Each loop does 40 adds, subtracts, and multiplies and 20 divides. Both `tdouble` and `tfloat` do 500 iterations, so a K & R compiler (which must convert *floats* to *doubles* before doing any arithmetic) should be slower on `tfloat` than on `tdouble`. The emerging ANSI standard no longer requires this promotion.

•For measuring basic integer arithmetic speed, we have `tint` and `tlong`. Both are identical, except that `tlong` has all its variables declared *long*, whereas `tint` uses integers. Each loop does 1,500 adds, 1,600 subtracts, 200 multiplies, and 200 divides. Note that `tint` does 1,500 loops, whereas `tlong` does only 1,000.

•`trig` does 12 calls each loop to `sin`, `cos`, and `tan` to measure the speed of trigonometric functions. Each function is called with constant .392699 and its multiples up through 5.890486.

As mentioned, we used a machine with a V30 processor rather than an 8088 or 8086. The effect of the V30 is strongest in three areas: memory access, multiplication, and string instructions. The 8086 has several addressing modes that vary from 2 to 15 clocks to compute. The V30 has the same addressing modes, but it takes only two clocks to compute all except

EditCheck syntax checks C source programs from within its multi-window editor.

EditCheck is a new kind of programmer's tool that provides an integrated C language source code programming platform. It improves your productivity, and reduces your frustration caused by making multiple cycles between a program editor and a compiler to manually locate and eliminate syntax, semantic, and other errors in your programs.

You can think of EditCheck as a powerful editing and program checking facility that works as a front-end companion to your C compiler.

EditCheck consists of an extensive C language source program checker, a window-oriented full-screen editor, a file access facility, and a context-sensitive on-line help facility, all united in a consistent, integrated environment platform that extensively uses a fast windowing system.

All EditCheck commands can be executed through single-key strokes, or thru a fast pop-up menu system, or by a combination of keystrokes and menu choices. All keystroke commands can be user reassigned to any keyboard key.

INTERACTIVE INTRA- & INTER-MODULE CHECKING

The heart of EditCheck's contribution is a powerful C language program checker which provides strong type checking from within the editing environment. The "lint-like" checker does a thorough evaluation of your source code and detects common errors and questionable practices, including many that most C compilers will overlook. The checker is based on "K & R" C specs plus Unix extensions and "lint" functions.

The checker runs interactively on part or all of a source program file, or group of program files. It opens a context window on the file where an error is detected and highlights the token which was being parsed when the error was detected. It also opens a message window with a descriptive error message, and presents a menu of options which you can take to correct the error.

The checker allows you to provide a program module list for complete checking or checking of unchecked modules. You can also check the modules in this list in batch mode if you desire.

Everest Solutions Inc.
3350 Scott Blvd., Bldg. 58
Santa Clara, CA 95054
(408) 986-8977

The source program checking capabilities in EditCheck will substantially increase your speed in achieving error-free C programs.

WINDOW-ORIENTED FULL-SCREEN EDITOR

EditCheck's editor allows you to edit text in the current window and to copy or move text within or between windows. The editor supports both horizontal and vertical scrolling, and allows you to use marks and zones, move the cursor to specified objects, search and replace (case sensitive or insensitive), change case of the text, control input mode, etc.

The editor uses a file paging scheme which allows you to edit and check modules larger than your available RAM memory. Any ASCII file may be read by the editor. Files may be inserted or appended to the current window file.

The editor is both key-command driven and menu driven, or mixed usage. Key-commands are fully user reassignable.

WINDOWS, FILES AND MORE

The EditCheck environment is window-oriented. You may have as many windows open at the same time as you wish. Windows may overlap or be tiled, at your option. You may switch back and forth between windows, and move or copy information between them.

The windows which you open may display different files or multiple different areas of the same file. You control the location and size of all user windows, and can save the contents of a window, hide it, bury it, close it, or show it.

Windows are also extensively used by the EditCheck system to build commands, display help, show a module list, display messages, show program context while checking, etc.

Access to some of the DOS file-oriented commands is also provided from within EditCheck.

A group of environment commands are available to change the coloring of windows (with a color graphics adapter and display), set the way you are notified of errors, and redefine the meaning of keys on the keyboard.

CONTEXT SENSITIVE HELP

Help is available to you in several ways. You may use a function key to get context sensitive help particular to where you are in the system. You may select the help index, and choose a topic of interest. You may also ask the help subsystem to search for a particular word of interest within the entire system. Display of current key-bindings is also available.

SYSTEM REQUIREMENTS

EditCheck requires an IBM PC/XT/AT or compatible, with 384K RAM, 2 flexible 5.25 inch disk drives or one flexible and one hard disk. DOS 2.0 or later.

HOW TO GET EDITCHECK

You can order today by mailing the coupon, or by phoning our toll free number. Send your check with the coupon, or we can ship COD (US only) or on your purchase order. No Credit Cards at this time. Outside USA please pay in US funds.

EditCheck

Everest Solutions, Inc.
3350 Scott Boulevard, Bldg. 58
Santa Clara, CA 95054

To Order Call:
1-800-621-0854 Ext. 923

or Send Coupon:

Please rush me EditCheck with 30 day money-back guarantee, and without copy-protection.

Price \$90.00 plus applicable charges:

(Please check items below):
___ CA. resid. add Sls. Tax: \$6.30
___ Ground ship add: \$3.00
___ USA Air Ship add: \$5.00
___ Outside USA Air Ship add: \$9.00
___ COD or Purch. Ord. add: \$5.00

Amount Enclosed: _____

or: COD requested (USA only) _____
or P.O. # _____

Co. Name _____

Name _____

Street _____

City _____

State _____ Zip _____

Day Phone _____

Circle no. 165 on reader service card.

Table and Figure Summary

All the tables and numbers can appear overwhelming at first, so we thought we should summarize the main points, table by table.

Table 1, page 36, indicates extras that come with a compiler to make it a programming environment.

Table 2, page 36, presents our memory models, which are used to provide bases of comparison for Table 3, page 38. Table 3 shows which compilers have which memory models, what they call them, and the options to invoke them.

Table 4, page 41, the dhrystone table, has some of our most surprising results. The "winners" here have to be the Datalight products. Although they do not support register variables, they produce the best results for no registers and lose to Microsoft C and IBM C by very little when registers are specified. We ran each compiler with the set of compiler options indicated in the flags column. Usually these flags told the compiler to relax the aliasing rules or to generate fast, rather than compact, code. See the section in the article on benchmarks for more information.

Table 5, page 42, shows execution times achieved using the small-memory model and is the heart of our benchmarks. The LC column represents the loop counts of the benchmarks. The section of the article on benchmarks provides more information about the loop counts. There are some significant holes in Table 5: Toolworks C could not compile our switches benchmark even after allowing more case label space. Eco-C88 did not produce an execution time for cpyblk, even though we think it ran. Overall, the Datalight products, High C, Microsoft C, IBM C, and Wizard C win in most of the categories. We are at a loss to explain the difference between Datalight C and Datalight C Developer's Kit for cpychr and think we did something to the latter to slow it down. Wizard C is weak in the library area, presumably because much of it is implemented in C rather than assembly. Whitesmiths C proves it has three register variables with a much

better time on regtest than reg2test. Mix C produced the fastest time for diskio but was disqualified because it got the wrong answer. Microsoft C and IBM C clearly won our switch2 and switch3 benchmarks but did so by optimizing away the whole switch, which was not our intent. A similar result on sieve and rsieve no longer necessarily means that register variables are not implemented. It could mean (and does for High C and Wizard C) that a compiler places variables in registers automatically. Note the variety of ways to allocate memory (memory) and do long arithmetic (tlong).

Table 6, page 43, the floating-point execution-time table, shows which compilers implement which floating-point options and how well they do so. Float 1, 2, and 5 are run without an 8087 at the indicated loop counts, and float 3, 4, 6, and 7 are run with an 8087 at ten times the indicated loop count. Also note that Toolworks C does all of its floating-point in single precision, not double. Microsoft C and IBM C implement all the floating-point options and do so fairly well. Lattice C also implements a large variety of floating-point options fairly well—especially float 1, the software-only option. Wizard C's use of the 8087 seems poor and may be a beta-test problem.

Table 7, page 44, shows how the memory models affect different types of code. The loop counts are on the line with the name of the benchmark. Pointer is most strongly affected by the size and type of data pointers, fibtest shows minor changes based on code pointer size, and sieve shows virtually no change at all.

Table 8, page 46, shows the compile and link times. The "average" benchmark is the average compile and link times for all the benchmarks run for time. (See Table 5 for the list.) We compiled doc1 through doc1000 with all the flavors reported for doc1000, but none of the compilers reported any significant difference between the flavors. The flavors are Min_clt, minimize compile and link time; Min_cs,

minimize code size; Min_xt, minimize execution time; and No_opts, no specific options. Here, Lattice C was not able to compile doc1000 for three flavors because we defined those flavors to use in-memory quad files, which would not fit for doc1000. Use No_opts when comparing across for Lattice C. Lattice C's link times are at an unfair advantage here because of the requirement of using LINK 3.0, which is faster than the linker used by all the other compilers that do supply a linker. DeSmet C is a very fast compiler with an equally fast linker. Wizard C does not seem very fast for small programs, but large programs compile much faster than average. Note that High C, the slowest for small programs, is in the top half of the list for doc1000.

The code-size table, Table 9, page 48, is not as varied as last year's. Hot C produced the smallest code sizes universally. Note that some compilers' Min_xt sizes are smaller than their Min_cs sizes. This is probably because of a difference in the way that function-entry code is handled and the assumption that more than one function will be linked. High C produced large files all the time though we did not use all the size-reducing mechanisms available.

Figure 1, page 50, represents the execution times of the pointer benchmark for memory models 2 and 5, which are the most common.

Figure 2, page 50, shows the execution times of the trig benchmark using the fastest software-only and hardware-assisted 8087 options.

Figure 3, page 50, shows compiler speeds, including compile times for both doc1 and doc1000, using options for each compiler to minimize compile time.

Figure 4, page 50, shows the dhrystone execution times with and without register variables.

The listings for the benchmarks used for this review begin on page 104.

one mode, which takes three clocks. The V30 also has a better shift mechanism, resulting in multiplies and shifts going faster. The string primitives run about twice as fast.

Use of the V30 affects our numbers but should not change the overall ordering of results. Benchmarks strongly affected are any that do multiplication (array, tint, tlong, and the software floating-point benchmarks) or string manipulations (strings and dhrystone).

Even though the V30 supports the 80186 instruction set, we did not instruct any of the compilers to use any instruction set except that of the 8086/8087.

Memory Models

Memory models are a continuing source of confusion for MS-DOS C compiler users. Here we describe, in our own terms, various memory models and their implementation. Then in Table 3, page 38, we indicate which compilers support which of our memory models.

Intel provided the definition, in general terms, of four memory models—small, compact, medium, and large. The distinctions between the models were in the size of each of two "spaces"—code space and data space. Either a space is less than 64K, or it is not. Small is small code and data, compact is small code with large data, medium is large code but small data, and large is large code and large data. As a variation of small, if

both code and data fit in a total of 64K, the program is called 8080 or tiny model.

For code space, the definition is adequate. If all code fits in 64K (called a segment when used in 8086 contexts—80386 protected mode segments are not limited to 64K), you use a small-code model. If all your code could not fit in a single segment, you need a large-code model.

For data space, however, Intel's standard is weak when applied to C. In C, there are three subspaces in the data space, all of which must be addressable by a single pointer type. If stack (where automatics are allocated), heap (where program-directed storage allocation takes place), and statics (including globals and usually string constants) all fit in the same

Extra Class	Aztec C	C86	Data-light C	Data-light Kit	DeSmet C	Eco-C88	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Micro-soft C	Mix C	Tool-works C	White-smiths C	Wizard C
DEBUG							1										
DEBUG SRCE	1				1					1	1		1			1	
DEBUG SYM								2									
DIFF	1		1	1						1							
EDITOR	3				1	1				2	1						
GREP	1		1	1						1	1	1					
MAKE	1		1	1				1		1			1				
MISC	4		7	7	5	1			1	4	3	7	1			2	
OBJ LIB	1	1			1		1	1	1	1	1		1			2	
OBJ UTIL	4						1	2	1	4	4		2	4		4	
PROFILER	1				3												
SRCE CNTRL		1			2												
SRCE LIB		1				1	1										1
SRCE LIB R	1														1		1

Table 1: Extras

Memory Model Number	Code Size	Static Size	Single Object Size	Total Data Size	Total Program Size	Memory Model Name
1	small	small	small	small	<64K	tiny
2	small	small	small	small	<128K	small
3	large	small	small	small	>64K	large code
4	small	small	small	large	>64K	fast large data
5	large	small	small	large	>128K	fast large
6	small	large	small	large	>64K	fast huge data
7	large	large	small	large	>128K	fast huge
8	small	small	large	large	>64K	slow large data
9	large	small	large	large	>128K	slow large
10	small	large	large	large	>64K	slow huge data
11	large	large	large	large	>128K	slow huge

Small means <64K. Large is >64K.

Table 2: Memory models and their attributes

segment, a small-data model works well. If, however, your total data requirements are more than 64K, things get complicated.

First, no compiler reviewed here supports a stack size of more than 64K. If you want to allocate a large array using the auto-storage class, or need to recurse more than 64K's worth of stack frames, you are out of luck.

Assuming you need a large-data model, pointers to data are 4 bytes. Given 4-byte pointers, there is no sense in restricting total heap space to anything less than all available memory, and none of these compilers do.

Statics, or any compile-time allocated data, are a different matter. Static objects can be referenced directly by name rather than through pointers as heap objects must be. If you have less than 64K of such objects, the compiler can have a segment register (*ds* usually) always pointing at the segment containing static data. We call this the medium-data model. If you need more than 64K of statics, then the compiler must continually load a segment register with the address of the segment containing the particular object referred to. This requires two extra instructions (*mov ax, seg* and *mov ds, ax*) for every direct access to a static object, which is normally a single instruction (*mov ax, [xyzzz]*). This is the large-data model.

So far we have described two code models (small and large) and three data models (small, medium, and large). With the addition of the tiny model, we have a total of seven memory models so far. These are enough if you have no single object (such as an array) larger than 64K in size. If any object is more than 64K (such as an array of 10,000 *doubles*, each 8 bytes), additional problems arise. A single 8086 segment is limited to 64K, so such an array must occupy more than one whole segment. This means that the compiler must allow for the adjustment of the segment part of an address as well as the offset part. This requires more code than simple offset manipulation needs, and this code can only be very slow.

Adding this variation to medium- and large-data models for both small and large code brings to 11 the total number of memory models. For a

concise description of the attributes of each of our 11 models, refer to Table 2, page 36. To see which memory models a compiler supports, look to Table 3. Table 3 lists, by compiler and our memory-model number, the compiler options needed to create the model, the compiler's name for the model, and which is the default.

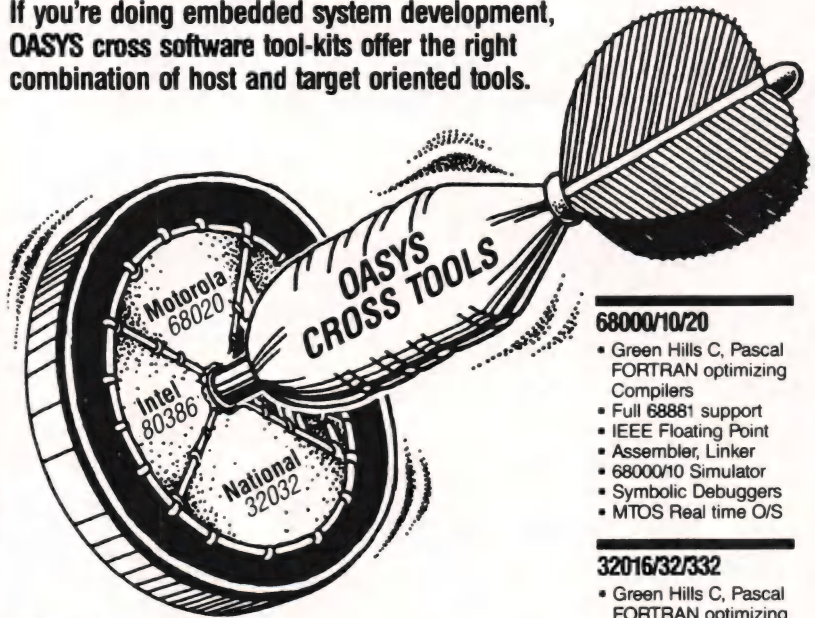
Lack of support for a particular memory model may not be fatal if you have a program needing that model. All even-numbered models (models with small code) can be com-

plied by a compiler supporting the number plus 1 (the same model with large code). Models 4 through 7 can be compiled by models 8 through 11, respectively, if you don't mind the extra overhead for 20-bit pointer arithmetic everywhere. Models 6, 7, 10, and 11 can be used in place of models 4, 5, 8, and 9, respectively, with a small penalty for accessing global data items. The Wizard compiler, for example, can compile and run a program needing less than 64K of code and more than 64K of statics,

OASYS

Cross Tools Hit All The Target Micros Of Your Choice

If you're doing embedded system development, OASYS cross software tool-kits offer the right combination of host and target oriented tools.



HOSTS: DEC VAX, Micro VAX, Sun, Apollo, Pyramid, Gould, and more! IBM PC, PC/XT, PC AT, compatibles, and OASYS PC Platform™

OASYS develops, supports, and enhances over 100 high quality, professional software development tools targeting popular 32-, 16-, and 8-bit micros.

HOT NEW PRODUCTS!

- Designer C++, based on AT&T's new C++ language, the next generation of C.
- OASYS PC Platform™ Co-Processor. Runs all OASYS tools on PC or Compatible.
- Ada on OASYS PC Platform™: VAX/780 power at 1/50th the cost.

We Specialize in: Cross/Native Compilers C, Pascal, FORTRAN, Ada, LISP — Assemblers/Linkers — Symbolic Debuggers — Simulators — Interpreters — Translators — Converters — Profilers — QA Tools — Design Tools — Comm. Tools — OS Kernels — Editors — Spreadsheets — Data Bases — VAX & PC Attached Processors and more

We Support: 680xx, 80x86, 320xx, 68xx, 80xx, and dozens more

68000/10/20

- Green Hills C, Pascal FORTRAN optimizing Compilers
- Full 68881 support
- IEEE Floating Point
- Assembler, Linker
- 68000/10 Simulator
- Symbolic Debuggers
- MTOS Real time O/S

32016/32/332

- Green Hills C, Pascal FORTRAN optimizing Compilers
- GENIX-compatible Assembler/Linker
- Symbolic Debuggers
- Compatibility with OASYS PC and VAX Co-Processor Boards

8086/186/286/386

- Optimizing 8086/186/286 and now 80386 Compilers
- Softprobe II Simulator and Debugger
- 80386 Structured Macro Assembler/Linker
- Full 8087 Support
- MTOS Real time O/S
- New 80386 compilers and assemblers coming soon!

Trademarks are acknowledged to: AT&T, Digital Equipment Corp., U.S. DOD Joint Program Office, Industrial Programming, Inc., IBM Corp., Motorola, Inc., Intel Corp., National Semiconductor Corp., XEL, Inc.

A DIVISION OF NEL **Oasys**

60 Aberdeen Ave., Cambridge, MA 02138 (617) 491-4180

Circle no. 254 on reader service card.

C COMPILERS

(continued from page 37)

but it must use a model allowing more than 64K of code.

To increase the performance of a program that has one or two large objects, with everything else able to fit

in a smaller data model, some compiler vendors implement what is known as mixed-model programming. With mixed-model programming, a small-model program can declare a pointer to be a *far* pointer—that is, a 4-byte pointer. IBM, Microsoft, Whitesmiths, and Wizard all im-

plement some form of mixed-memory-model support. Whitesmiths' compiler can never automatically deal with objects larger than 64K. Wizard's can but on a source-file-by-source-file basis—that is, all *far* pointers in a module are manipulated using the same rules about the

#	Aztec C		C86		Datalight C
1 -o	tiny			-mc	tiny
2	small		small	-ms	small
3 +lc	large code				
4 +ld	large data				
5 +l	large	-b	large		
#	Datalight Kit		DeSmet C		Eco-C88
1 -mc	com				
2 -ms	S		small		small
3 -mp	P				
4 -md	D				
5 -ml	L	-b	large		
#	High C		Hot C		IBM C
1					
2	small		small	-AS	small
3	medium			-AM	medium
4	compact				
5	big	-b	large		
6					
7	large			-AL	large
8					
9					
10					
11				-AH	huge
#	Lattice C		Microsoft C		C Programming System
1					
2 -mS	S	-AS	small	-vsmall	small
3 -mP	P	-AM	medium		
4 -mD -s	D				
5 -mL -s	L			-vlarge	large
6		-AC	compact		
7		-AL	large		
8 -mD	D				
9 -mL	L				
10					
11		-AH	huge		
#	Lets' C		Mix C		Toolworks C
1					
2	small				small
#	Whitesmiths C		Wizard C		Our Model Name
1 -dcom	8080	-mtf	tiny 16		tiny
2 -dmods	s	-msf	small 16		small
3 -dmodp	p	-mmf	medium 16		large code
4 -dmodd	d	-mcf	compact 16		fast large data
5 -dmodf	f	-mlf	large 16		fast large
6					fast huge data
7		-mhf	huge 16		fast huge
8		-mc	compact 20		slow large data
9		-ml	large 20		slow large
10					slow huge data
11		-mh	huge 20		slow huge

Table 3: Support for memory models

C spoken here...

High C™

Do you want to use a C compiler that

- was chosen by Ashton-Tate for implementing dBASE III® Plus
- was well rated in *Computer Language*, Feb. 86 and *Dr. Dobbs Journal*, August 86
- "would have saved me three weeks of porting time had I had High C instead of Microsoft's new C"
Mike LeBlanc, compiler developer, Sky Computers
- "is the only C compiler for the IBM PC capable of compiling NYU's Ada/Ed compiler"
Dave Shields, research scientist, New York Univ.
- has a complete run-time library
- has structure assignment, **enum**, **void**...
- supports nested functions as in Pascal
- supports pcc and full K&R C plus some latest, nifty extensions from the new ANSI-proposed C standard
- "saved 15% of code over five large modules of MultiMate relative to Lattice C"

David Beauchesne, Multimate International

Pascal spoken here...

Professional Pascal™

Do you want to use a Pascal compiler that

- was chosen by Lifetree Software, Inc., for implementing Volkswriter Deluxe™
- was well rated in *Computer Language*, May 86, p. 90: "The clear choice for large-scale programming projects..."
- serves as a systems and applications language at CAD/CAM giant Daisy Systems Corporation
- has 8-, 16-, and 32-bit integers; sets up to 64K bits
- has varying-strings of up to 64K characters
- has a full-fledged C macro preprocessor
- has many run-time library additions: UNIX™-like I/O, multiple heaps, interrupts, . . .
- has all the bit-pushing operators of C
- has many more extensions, getting you half way to Ada® for a non-Ada price
- is the "howitzer" of Pascals and "could well be the most powerful Pascal compiler ever implemented on a microcomputer" *PC Magazine*, Oct. 29, 1985, p. 144

Power Tools



for

Power Users

Each compiler • generates **superb code**, with optimizations such as common-subexpression elimination and cross-jumping • sports no less than **five memory models** for the 8086 (Small, Compact, Medium, Big, and Large) • supports a unique implementation of register variables • supports the **8087/80287** in native mode, or **emulates** • supplies **three** floating-point formats • generates special instructions for the 80186/286 • generates code that runs in **80286 protected mode** • gives you **hundreds of error and warning messages**, helping you find those subtle bugs *before* you need a debugger's help • lets you **overlay data** as well as code (when used with PLINK86), for substantial space savings • lets you write **interrupt routines** directly in high-level language • lets you get "close to the machine" with built-in move/scan/compare operations • is supported by equivalent **resident and cross** compilers for the 80286 (Xenix, Concurrent DOS 286), 68010/20/68881 (UNIX V & 4.2, GEM-DOS™), 32032 (UNIX 4.2), VAX (UNIX 4.2, VMS), IBM 370,... • contains a **multi-modular cross-referencer** • produces ROM-able code for embedded applications • can talk to those **other languages** by those other vendors • gives you **64K run-time stack** space that can be shared with the heap • is endowed with an **amazing number of pragmas** (compiler controls) for customization to your application • has a compiler start-up **profile** • supports direct access to MS-DOS; library supports DOS 3.X file-sharing • generates symbolic debugger information for use with all known MS-DOS debuggers • allows **exec-ing** subprocesses • was designed for professional software developers, not hobbyists • comes with great technical support by a company that specializes in compilers • comes with extensive **typeset documentation** • and *more...* call or write for your information packet today...

Not recommended for casual use, but for applications needing **industrial-strength tools**, contact



INCORPORATED

903 Pacific Avenue, Suite 201, Santa Cruz, CA 95060
(408) 429-6382 (429-META), TELEX: 4930879

Abroad:

ABC Software, The Netherlands
Microsoft, Tokyo
Grey Matter, United Kingdom
Buchdata, Frankfurt

Since 1979

High C: \$495

—on any MS/PC-DOS system—

Professional Pascal: \$595

OEMs: Contact us about porting our professional compilers to your systems.

TWS: Professional Compiler Developers and competitors, ask about our Translator Writing System compiler toolbox; see the review in *Computer Language*, December, 1985.

DOS Helper™: Need UNIX-like utilities to enhance MS-DOS? Ask about our powerful tools for \$49.95—included free with MetaWare compilers: FIND, TAIL, MV, LS, CAT, UNIQ, FGREP, and WC.

• MetaWare, High C, Professional Pascal, and DOS Helper are trademarks of MetaWare Incorporated • Other trademarks and their owners are: UNIX—AT&T, dBASE III—Ashton-Tate, Volkswriter Deluxe—Lifetree Software, GEM-DOS—Digital Research, Ada-DoD. © 1986 MetaWare.

C COMPILERS

(continued from page 38)

maximum size of a single object. IBM and Microsoft offer two different kinds of *far* pointers—*far* and *huge*. *Far* implies less than 64K per object, and *huge* implies more than 64K.

To test memory-model performance, we ran array, pointer, sieve, fibtest, and memory under all the available memory models. We list the results produced in these tests in Table 5, page 42.

Floating-Point Options

Almost as complicated as memory models, floating-point options are another point of distinction between various compiler vendors. Programs that use floating point can be created in five different ways.

The fastest, smallest option is to have in-line 8087 instructions without the presence of an emulation library to handle the machine without an 8087.

Option 2 is to put 8087 instructions in-line and use an emulation library

in case the machine doesn't have a real 8087. In reality, this option does not actually put 8087 instructions in-line. Where 8087 instructions belong, software interrupts are placed instead. When the interrupt occurs, if an 8087 is present, the interrupt instruction is replaced by the actual instruction and executed.

Next, where floating-point operations are needed, calls are made to a library. The kind of library linked in determines which of the last three floating-point options is in use.

The library may require the presence of an 8087 to execute. Or it may be a "sensing" library, testing a word set at start-up indicating the presence or absence of an 8087 and branching to or around 8087 instructions. Finally, it could be a pure software library, which would probably be faster than an emulation library on a non-8087-equipped machine but slower than a library using an 8087.

We ran *dfuncrct*, *prtf*, *tfloat*, *tdouble*, and *trig* using all the supported 8087 modes for each compiler. Results from these benchmarks are shown in Table 6, page 43.

Compiler-Specific Comments

The raw data are only part of the story. In the following, we attempt to summarize the important remaining facts and observations about each compiler.

C86

Computer Innovations is one of the old names in MS-DOS C compilers. It has been marketing its C86 C compiler almost as long as has Lattice. The compiler package does not seem much changed from last year, and we are told that a major enhancement is due soon.

The compiler provides for small/small-and large/large-memory-model programming. (A large-code/small-data model was in beta test but was not reviewed.) It provides 8087 support via an option to generate in-line 8087 code and a separate library (good), but there is no way to build a program that will use an 8087 but also run without one (bad). There is an option to generate code for an 80186 or 80286 processor. Full source code for the libraries is provided in archive form (along with an archive maintenance program). There is

Programmer Essentials

"Offers many capabilities for a reasonable price"

W. Hunt, PC Tech Journal

"I highly recommend the  UTILITY LIBRARY"

D. Deloria, The C Journal

ESSENTIALS

200 functions: video, strings, keyboard, directories, files, time/date and more. Source code is 95% C. Comprehensive manual with plenty of examples. Demo programs on diskette. Upgrade to THE C UTILITY LIBRARY for \$95.

\$100

THE UTILITY LIBRARY

Thousands in use world wide. 300 functions for serious software developers. The C ESSENTIALS plus "pop-up" windows, business graphics, data entry, DOS command and program execution, polled async communications, sound and more.

\$185

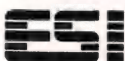
ESSENTIAL GRAPHICS

\$250

Fast, powerful, and easy to use. Draw a pie or bar chart with one function. Animation (GET and PUT), filling (PAINT) and user definable patterns. IBM color, IBM EGA and Hercules supported (more soon). NO ROYALTIES. Save \$50 when purchased with above libraries.

Compatible with Microsoft Ver. 3, Lattice, Aztec, Mark Williams, CI-C86, DeSmet, and Wizard C Compilers. IBM PC/XT/AT and true compatibles.

Compiler Packages: Microsoft C - 319, Lattice or CI-C86 compilers \$329. Save \$40 - \$50 when purchasing compiler and library combinations. Specify C compiler and version number when ordering. Add \$4 for UPS or \$7 for UPS 2-day. NJ residents add 6% sales tax. Visa, MC, Checks, PO's.



ESSENTIAL SOFTWARE, INC

P.O. Box 1003 Maplewood, NJ 07040 914/762-6605

Circle no. 138 on reader service card.

even an access library for TopView included in the package—something no other vendor provides (if anybody cares).

Installing this compiler is a pain, mainly because all the distribution files come in a compressed format (all four disks' worth). You have to un-squeeze everything before you do anything else, an unpleasant operation on a dual-floppy machine. The unsqueeze program did not detect a full target disk at one point. Computer Innovations supplies a library that will work with all DOS versions, including 1.0, and a separate (and preferable) library that requires DOS 2.0 or later. We used that one, as will most users.

Computer Innovations is the only vendor besides Software Toolworks that does not provide a driver program at least for the compile/assemble process. Each pass of the compiler (there are four) must be invoked separately from the command line or from a batch file. All options are given to the first pass. There is an unusual optimization option that lets you specify how many hundreds of bytes of code space per *switch* statement can be spent building a jump table, compared to the space a linear search would take. We used *-j5* as a reasonable trade-off for the execution-time benchmarks.

The documentation consists of about 330 8.5×11-inch pages in a reasonable binder (larger than IBM size). It includes an extremely detailed table of contents that doubles as a quick reference for the library, and it also has a lengthy index and a glossary section. The library documentation is the best part of the manual. It is clear and well organized, with one function per page and in alphabetical order, and includes a fair number of examples. The rest of the manual seems to be a hodgepodge of technical information presented in bits and snatches, with poor overall organization. There is no language reference—you are directed to K & R for that. Worse yet, there is not enough information about implementation-specific details such as whether *chars* signs extend when promoted to *ints*. One page of information appears under the heading "Language information," and a few other tidbits are scattered about, and that's it. Our

impression is that this package is aimed at the hobbyist/hacker community.

According to the read.me file, several bugs have been fixed in the last few releases of the compiler. There

are still a couple left, though. For instance, the compiler will be fooled by what looks like an opening comment delimiter inside a quoted string unless it is escaped. In other words, `printf("- /* -");` gives a compiler er-

Compiler Name	Without Registers		With Registers		Flags
	Time	Rating	Time	Rating	
Aztec C	53.7	931	50.8	984	+a +F
C86	91.9	544	91.9	544	-j5
Datalight C	44.7	1118	44.9	1113	none
Datalight Kit	44.8	1116	44.7	1118	none
DeSmet	61.3	813	61.3	813	none
Eco—C88	60.3	829	60.3	829	none
High C	54.0	925	54.0	925	none
Hot C	62.7	797	57.0	877	none
IBM C	46.7	1070	44.6	1121	-Oat -Gs
Lattice C	62.7	797	62.6	798	-ms -cw
Mix C	607.4	82	607.4	82	none
Microsoft C	46.6	1072	44.4	1126	-Oat -Gs
C Prog. System	85.8	582	80.6	620	none
Let's C	85.9	582	80.6	620	none
Toolworks C	CNC	CNC	CNC	CNC	typedef
Whitesmiths C	97.1	514	93.2	536	none
Wizard C	63.8	783	63.6	786	-a -G -o -z

Table 4: Dhrystone results


Brand New From Peter Norton A PROGRAMMER'S EDITOR

that's *lightning fast* with the *hot* features programmers need

only
\$50

Direct from the man who gave you *The Norton Utilities*, *Inside the IBM PC*, and the *Peter Norton Programmer's Guide*.

THE NORTON EDITOR™



"This is the programmer's editor that I wished I'd had when I wrote my *Norton Utilities*. You can *program your way to glory* with *The Norton Editor*."

Peter Norton



Easily customized, and saved
Split-screen editing
A wonderful condensed/outline display
Great for assembler, Pascal and C

Peter Norton Computing, Inc., 2210 Wilshire Boulevard,
Santa Monica, CA 90403, 213-453-2361. Visa,
Mastercard and phone orders welcome.

The Norton Editor™ is a trademark of Peter Norton Computing, Inc. © 1986 Peter Norton Computing.

Circle no. 243 on reader service card.

C COMPILERS

(continued from page 41)

ror, but `printf(“- \/* -”);` works fine. The documentation warns you, though. Also, we got a meaningless error message when we attempted a static function declaration (not definition). The declaration `static int func0();` produced the error message “get size error for 103.”

DeSmet C

C Ware Corp. has been around for quite a while now, and its DeSmet C development package has attracted many devoted fans over the years. This is because it has always provided a fairly complete implementation of C along with a source-level debugger and several other goodies for an affordable price. It includes a text editor (becoming more common) as well as an execution profiler (uncommon).

The latest upgrade to the DeSmet C compiler adds support for large/large-memory-model programming and the 8087 chip. We tested a beta version of this release, but we did not receive the corresponding upgraded documentation.

DeSmet C uses its own object format; Microsoft object-format support is available as an extra-cost option. Overlays are supported in the small-memory model. The compiler supports `enum` and structure assignment and `pass/return`.

The large-model version of the compiler comes on three disks. It has no installation guide except for a discussion of `CONFIG.SYS` and the use of RAM disks (a RAM-disk driver comes with the package). There is a lot of extra stuff on the distribution disks, so the compiler, assembler, linker, and libraries can fit easily on one disk. It has support for an 8087 on a take-it-or-leave-it basis, and it has a

software-only library and an 8087-only library. The installation guide tells you to pick the one you want (both contain all the other functions as well as the floating-point functions), name it `csdio.s`, and delete the other one. You can tell the linker (bind) to search a specific library by name, but it automatically scans `csdio.s` in any case.

The manual comes in an IBM-size, D-ring binder with pages somewhat wider than usual. The writing style is terse, and it is sometimes difficult to find the information you are looking for. It has no index but has a reasonable table of contents. Pages are numbered by section, making it easier to insert updates but harder to flip to a desired page.

The documentation has no language reference, but it has a brief summary of differences from the K & R standard. If you buy this package, you will need some other book to

Benchmark	LC	Aztec C	C86	Data-light C	Data-light Kit	DeSmet C	Eco-C88	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Microsoft C	Mix C	Tool-works C	White-smiths C	Wizard C
array	1500	37.2	78.9	53.1	53.1	78.2	78.0	61.8	37.9	54.9	82.7	82.6	35.1	37.9	121.0	142.0	71.0	59.3
autotst	150	45.2	47.8	47.8	47.7	47.8	47.7	55.7	45.2	47.7	47.7	47.7	36.9	45.2	83.5	66.6	47.0	33.3
cpyblk	15	21.0	40.5	23.0	23.7	2.9	—	25.3	5.7	41.8	19.4	19.7	8.3	5.2	28.2	87.4	12.9	27.2
cpychr	15	15.4	15.3	13.0	13.7	51.9	26.3	22.8	17.7	37.4	22.6	23.0	76.6	17.3	46.7	54.6	35.9	18.9
diskio	350	9.0	7.8	24.1	24.0	7.2	8.4	6.2	9.6	8.6	7.5	7.5	9.0	9.6	1.4	8.8	18.3	8.2
fibtest	18	34.1	33.7	30.3	30.3	31.9	32.1	42.9	37.0	34.3	39.3	39.4	38.3	37.1	178.9	38.3	40.2	35.2
fillscr	12	32.1	16.7	28.8	28.8	14.7	19.3	30.1	13.0	32.3	13.8	13.7	13.0	12.9	45.1	35.4	13.6	29.1
funcov0	10000	47.3	47.6	29.8	29.9	46.4	29.9	59.9	45.9	46.5	72.5	72.6	49.4	45.8	477.6	34.3	58.3	37.0
funcov1	10000	71.4	71.7	65.9	65.9	68.6	68.6	94.4	68.5	68.2	95.3	95.2	65.9	68.5	499.8	46.7	81.4	54.1
funcov2	10000	85.2	85.3	81.9	81.8	83.2	82.1	108.6	82.0	80.9	109.6	109.7	79.5	82.1	521.5	60.9	100.8	62.0
funcov3	10000	98.4	99.3	96.5	96.4	99.6	97.1	128.0	96.8	99.8	124.6	124.6	95.4	96.7	544.0	81.0	109.8	79.1
looptst	500	36.7	38.8	38.9	39.0	39.8	39.0	45.7	36.7	41.0	38.9	38.9	17.5	25.8	60.8	43.6	38.9	15.4
memory	500	45.3	188.9	56.7	56.7	186.4	170.4	59.6	32.0	295.6	1211.0	1211.0	384.5	31.4	259.6	401.7	109.4	77.6
optimize	100	17.7	20.1	15.5	15.5	21.7	20.9	24.7	14.4	7.8	20.4	20.5	9.1	10.8	27.9	20.3	20.7	9.3
pointer	1500	31.0	40.7	38.7	38.7	39.5	38.7	38.8	31.0	38.7	43.5	43.6	33.0	30.9	93.0	71.1	42.4	30.7
reg2test	150	34.8	47.7	47.7	47.7	47.7	47.7	38.6	33.3	47.7	35.9	35.9	38.1	33.3	84.8	66.5	36.5	33.3
regtest	150	33.6	47.8	47.7	47.8	47.8	47.7	38.5	33.3	47.8	35.9	35.9	36.8	33.3	84.8	66.5	30.0	33.4
rsieve	140	36.8	67.7	60.1	60.0	63.2	64.1	37.4	34.6	59.8	36.7	36.7	37.0	34.6	110.1	68.2	37.4	34.3
scroll	12	48.0	32.5	44.7	44.9	30.6	35.2	46.1	29.0	47.9	29.7	30.0	29.3	29.2	45.3	51.1	29.8	44.6
sieve	140	61.2	67.7	60.0	60.0	63.2	64.1	71.3	60.2	59.9	63.2	63.2	37.1	60.1	96.2	86.8	68.1	44.4
stattst	150	49.6	64.1	50.6	50.7	61.9	53.3	49.6	49.6	53.3	53.3	53.3	53.1	49.6	100.6	82.3	52.6	46.4
strings	1000	5.8	10.9	3.8	3.8	9.2	13.8	7.7	4.1	10.2	15.7	15.7	4.2	3.9	152.1	36.1	15.1	8.9
switch1	1000	8.6	14.0	5.7	5.8	10.0	35.5	8.5	7.1	6.2	5.9	6.0	5.5	7.1	85.6	—	13.8	6.3
switch2	1000	8.5	8.8	5.6	5.7	8.0	8.3	7.6	3.9	6.1	5.8	5.8	5.2	3.8	18.6	—	13.5	4.8
switch3	1000	17.1	11.9	11.5	11.6	44.8	12.0	11.3	3.7	31.7	27.0	27.0	8.1	3.7	30.6	—	28.7	24.8
tint	1500	15.2	16.8	14.6	14.7	16.3	16.1	18.6	14.5	16.3	15.1	15.1	14.4	14.5	23.2	27.9	15.1	14.0
tlong	1000	75.8	129.6	49.6	49.5	100.6	158.4	66.5	52.9	58.0	114.6	114.5	55.8	50.3	147.7	167.2	54.6	82.5
Winners	0	0	5	4	4	1	1	1	5	1	0	0	5	6	0	2	1	9
10%	6	0	3	4	4	2	1	2	8	0	6	6	5	7	0	1	5	2
Total	6	0	8	8	8	3	2	3	13	1	6	6	10	13	0	3	6	11

Boldfaced entries represent the top scores for each benchmark. Italicized entries came within 10% of the top score. The three rows at the bottom summarize the number of winning and runner-up scores for each compiler.

Table 5: Regular EXE time

go with it. The library reference is organized by category, with the category name in LARGE type at the top of each page. An alphabetical list of functions at the front tells you what category name to look under for the function definition. Several functions are defined on each page, à la Unix, with a usage summary that takes some getting used to even if you like Unix documentation.

Datalight C

Datalight distributes two versions of its C compiler. The personal version, called the Datalight C compiler, supports only the small/small-memory model and does not include source code for the library. Support for an 8087 is provided, via a sensing library. It has been completely rewritten since last year.

The entire package is distributed

on one sealed disk, and the license agreement is visible on the outside of the package. The agreement is short and reasonable, requiring only that the package not be used by more than one person at a time. We think that means you can take it with you to work, and install it on your new computer, without violating your agreement. This type of agreement is starting to show up more often, re-

Benchmark	LC	Aztec C	C86	Datalight C	Datalight Kit	DeSmet C	Eco-C86	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Microsoft C	Mix C	Toolworks C	White-smiths C	Wizard C
atox	100																	
Float_1		31.3	22.6	--	--	17.1	--	--	3.5	55.4	19.8	19.8	--	3.3	--	12.8	12.3	--
Float_2		32.1	--	130.9	130.9	--	6.0	1590.0	4.1	55.3	--	--	9.8	4.2	--	--	--	7.9
Float_3		45.0	--	128.6	128.6	--	--	83.7	21.4	565.4	--	--	40.6	21.3	--	--	--	49.5
Float_4		44.1	--	--	--	21.4	--	83.7	21.5	565.4	--	--	--	21.3	--	121.6	--	--
Float_5		--	--	--	--	--	--	--	4.2	--	--	--	--	4.2	--	--	--	--
Float_6		--	--	--	--	--	--	--	21.5	565.7	--	--	--	21.3	--	--	--	--
Float_7		--	35.6	--	--	--	--	--	21.4	565.7	68.9	--	33.9	21.4	--	--	33.2	49.6
dfuncrct	250																	
Float_1		56.5	51.1	--	--	54.7	--	--	24.3	6.2	12.2	12.2	--	24.3	48.4	5.8	14.2	--
Float_2		62.2	--	6.4	6.4	--	9.5	325.7	85.4	6.3	--	--	11.3	86.4	--	--	--	19.4
Float_3		367.6	--	63.9	63.9	--	--	98.7	15.9	62.8	--	--	113.6	158.6	--	--	--	193.4
Float_4		298.0	--	--	--	234.8	--	98.8	15.8	62.7	--	--	--	158.7	--	58.1	--	--
Float_5		--	--	--	--	--	--	--	82.9	--	--	--	--	83.7	--	--	--	--
Float_6		--	--	--	--	--	--	--	125.3	236.1	--	--	--	125.3	--	--	--	--
Float_7		--	243.8	--	--	--	--	--	125.2	236.1	116.1	--	113.6	125.2	--	--	129.5	147.1
prtf	12																	
Float_1		49.3	46.3	--	--	34.5	--	--	29.6	56.4	32.3	32.5	--	29.8	56.6	52.2	37.1	--
Float_2		49.5	--	--	--	--	35.2	85.1	29.7	56.1	--	--	35.7	29.9	--	--	--	49.5
Float_3		477.5	--	--	--	--	--	472.1	292.5	564.9	--	--	333.2	295.5	--	--	--	481.0
Float_4		477.8	--	--	--	321.3	--	472.0	292.8	565.5	--	--	--	295.4	--	518.9	--	--
Float_5		--	--	--	--	--	--	--	29.8	--	--	--	--	29.8	--	--	--	--
Float_6		--	--	--	--	--	--	--	292.4	564.4	--	--	--	295.4	--	--	--	--
Float_7		--	351.9	--	--	--	--	--	292.4	565.6	308.1	--	326.8	295.6	--	--	313.8	481.5
tdouble	500																	
Float_1		41.8	86.6	--	--	39.4	--	--	21.3	32.7	17.2	17.2	--	21.3	98.8	8.9	118.0	--
Float_2		43.7	--	36.8	36.8	--	5.8	396.2	49.6	33.4	--	--	56.4	50.1	--	--	--	31.6
Float_3		70.6	--	47.2	47.2	--	--	27.5	35.3	53.6	--	--	209.6	35.7	--	--	--	75.3
Float_4		60.5	--	--	--	50.8	--	27.5	35.2	53.7	--	--	--	35.7	--	47.4	--	--
Float_5		--	--	--	--	--	--	--	50.1	--	--	--	--	50.6	--	--	--	--
Float_6		--	--	--	--	--	--	--	27.2	27.8	--	--	--	27.2	--	--	--	--
Float_7		--	29.1	--	--	--	--	--	27.2	27.8	26.8	--	26.9	27.2	--	--	29.1	29.1
tfloat	500																	
Float_1		41.7	90.8	--	--	57.2	--	--	11.9	46.2	19.2	19.1	--	12.0	100.4	8.9	117.6	--
Float_2		43.5	--	64.3	64.3	--	11.2	337.7	44.8	46.7	--	--	46.2	45.1	--	--	--	38.4
Float_3		67.1	--	114.3	114.3	--	--	24.8	34.9	114.0	--	--	212.8	34.6	--	--	--	136.3
Float_4		56.8	--	--	--	48.1	--	--	34.9	114.0	--	--	--	34.6	--	47.4	--	--
Float_5		--	--	--	--	--	--	--	45.1	--	--	--	--	45.5	--	--	--	--
Float_6		--	--	--	--	--	--	--	24.1	30.5	--	--	--	24.1	--	--	--	--
Float_7		--	26.4	--	--	--	--	--	24.1	28.0	23.8	--	24.2	24.1	--	--	26.9	150.4
trig	100																	
Float_1		44.7	72.1	--	--	47.5	--	--	26.7	10.4	24.5	24.5	--	26.7	148.8	17.4	104.6	--
Float_2		46.2	--	53.4	53.4	--	7.9	650.5	41.5	42.6	--	--	47.2	42.2	--	--	--	42.8
Float_3		98.5	--	98.3	98.3	--	--	56.7	16.4	15.8	--	--	17.9	16.4	--	--	--	154.1
Float_4		81.8	--	--	--	63.1	--	--	16.4	15.8	--	--	--	16.4	--	92.7	--	--
Float_5		--	--	--	--	--	--	--	41.5	--	--	--	--	42.2	--	--	--	--
Float_6		--	--	--	--	--	--	--	16.0	17.0	--	--	--	15.9	--	--	--	--
Float_7		--	13.8	--	--	--	--	--	15.9	16.9	9.7	--	11.3	15.9	--	--	25.6	154.1

Table 6: Floating-point EXE time

placing the ridiculously restrictive licenses people are used to ignoring. This trend should be encouraged.

Batch files are provided to automate hard-disk and floppy-disk installation, but they are hardly necessary with such a small package. The documentation includes sample sessions and describes all environment variables used. Source code is included for a few Unix-type utility pro-

grams—most notably fgrep and diff—and a simple make utility is also included (executable only).

Datalight gives you the same manual regardless of which version of the compiler you buy. Thus, if you buy the personal version, you should be prepared to ignore references throughout the manual to features you don't actually have. The quantity of documentation is below average (210 IBM-size pages), but the quality is good. The material is readable and covers the compiler's features well

for the most part. The manual has sections describing the compiler's optimization strategies and aliasing assumptions (so you can write faster code), and implementation-defined behavior is documented. It has a table of contents and an index.

A bug prevented compilation of the prtf benchmark. The version of the compiler we received cannot compile a code fragment such as:

```
float f;
foo(-f, f, -f);
```

Benchmark	LC	Aztec C	C86	Datalight C	Datalight Kit	DeSmet C	Eco-C88	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Microsoft C	Mix C	Toolworks C	White-smiths C	Wizard C
array	1500																	
MemMdl_2		37.2	78.9	53.1	53.1	78.2	78.0	61.8	37.9	54.9	82.7	82.6	35.1	37.9	121.0	142.0	71.0	59.3
MemMdl_3		37.3	—	—	53.1	—	—	—	38.0	54.8	—	—	49.0	38.0	—	—	67.4	59.3
MemMdl_4		40.6	—	—	60.6	—	—	—	—	54.8	—	—	49.0	—	—	—	82.6	59.4
MemMdl_5		40.6	103.2	—	60.6	—	—	—	—	54.8	102.2	—	49.0	—	—	—	87.6	59.3
MemMdl_7		—	—	—	—	—	—	—	37.9	—	—	—	49.0	37.9	—	—	—	59.4
MemMdl_8		—	—	—	—	—	—	—	—	54.8	—	—	—	—	—	—	—	251.0
MemMdl_9		—	—	—	—	—	—	—	—	54.8	—	—	—	—	—	—	—	277.8
MemMdl_11		—	—	—	—	—	—	—	37.9	—	—	—	—	38.0	—	—	—	278.9
fibtest	18																	
MemMdl_2		34.1	33.7	30.3	30.3	31.9	32.1	42.9	37.0	34.3	39.3	39.4	38.3	37.1	178.9	38.3	40.2	35.2
MemMdl_3		37.2	—	—	33.5	—	—	—	40.2	37.9	—	—	39.3	40.2	—	—	39.3	39.4
MemMdl_4		34.1	—	—	30.8	—	—	—	—	34.4	—	—	35.3	—	—	—	40.2	35.3
MemMdl_5		37.2	36.4	—	33.8	—	—	—	—	37.9	43.0	—	39.3	—	—	—	39.3	39.4
MemMdl_7		—	—	—	—	—	—	—	40.1	—	—	—	45.4	40.2	—	—	—	45.2
MemMdl_8		—	—	—	—	—	—	—	—	34.4	—	—	—	—	—	—	—	35.2
MemMdl_9		—	—	—	—	—	—	—	—	38.0	—	—	—	—	—	—	—	39.4
MemMdl_11		—	—	—	—	—	—	—	40.2	—	—	—	—	40.1	—	—	—	45.2
memory	500																	
MemMdl_2		45.3	188.9	56.7	56.7	186.4	170.4	59.6	32.0	295.6	1211.0	1211.0	384.5	31.4	259.6	401.7	109.4	77.6
MemMdl_3		46.8	—	—	57.3	—	—	—	32.8	306.9	—	—	408.9	32.6	—	—	103.5	79.8
MemMdl_4		122.3	—	—	138.5	—	—	—	—	1011.3	—	—	419.7	—	—	—	—	1028.3
MemMdl_5		125.3	1288.9	—	145.0	—	—	—	—	1064.3	2108.1	—	434.3	—	—	—	—	1050.5
MemMdl_7		—	—	—	—	—	—	—	37.2	—	—	—	445.0	37.7	—	—	—	322.0
MemMdl_8		—	—	—	—	—	—	—	—	1051.4	—	—	—	—	—	—	—	1079.2
MemMdl_9		—	—	—	—	—	—	—	—	1102.3	—	—	—	—	—	—	—	1107.2
MemMdl_11		—	—	—	—	—	—	—	44.2	—	—	—	—	44.2	—	—	—	376.6
pointer	1500																	
MemMdl_2		31.0	40.7	38.7	38.7	39.5	38.7	38.8	31.0	38.7	43.5	43.6	33.0	30.9	93.0	71.1	42.4	30.7
MemMdl_3		30.9	—	—	38.7	—	—	—	31.0	38.7	—	—	37.6	31.0	—	—	41.2	30.7
MemMdl_4		46.5	—	—	62.0	—	—	—	—	60.3	—	—	48.6	—	—	—	68.3	54.2
MemMdl_5		46.4	—	—	61.9	—	—	—	—	60.3	75.9	—	48.6	—	—	—	69.8	54.2
MemMdl_7		—	—	—	—	—	—	—	49.0	—	—	—	48.7	49.1	—	—	—	54.1
MemMdl_8		—	—	—	—	—	—	—	—	185.8	—	—	—	—	—	—	—	267.1
MemMdl_9		—	—	—	—	—	—	—	—	191.7	—	—	—	—	—	—	—	278.6
MemMdl_11		—	—	—	—	—	—	—	81.3	—	—	—	—	81.3	—	—	—	275.9
sieve	140																	
MemMdl_2		61.2	67.7	60.0	60.0	63.2	64.1	71.3	60.2	59.9	63.2	63.2	37.1	60.1	96.2	86.8	68.1	44.4
MemMdl_3		61.2	—	—	60.0	—	—	—	60.2	59.9	—	—	59.3	60.2	—	—	67.5	44.4
MemMdl_4		61.1	—	—	59.5	—	—	—	—	59.9	—	—	59.2	—	—	—	75.1	44.4
MemMdl_5		61.2	110.0	—	59.5	—	—	—	—	59.9	76.5	—	59.2	—	—	—	75.5	44.3
MemMdl_7		—	—	—	—	—	—	—	60.1	—	—	—	59.2	60.2	—	—	—	44.4
MemMdl_8		—	—	—	—	—	—	—	—	59.9	—	—	—	—	—	—	—	45.5
MemMdl_9		—	—	—	—	—	—	—	—	59.9	—	—	—	—	—	—	—	45.5
MemMdl_11		—	—	—	—	—	—	—	60.2	—	—	—	—	60.1	—	—	—	45.7

Table 7: Memory-model EXE time

THE PROGRAMMER'S SHOP

31 Day
RISK-FREE TRIAL
on any product in this ad.

C Programmers: 8 Ways to Increase Productivity

Pascal to C Translation Easily and at Low Cost with The Translator (Pascal to C)

If you like Pascal's English-like syntax but want C's portability, speed, and control, or if you want to make a permanent switch to C at a very low cost, Milton Brown's Pascal to C translator is for you.

Take Jensen and Wirth standard Pascal and produce K&R standard C code. Any non-standard Pascal syntax is passed directly to the C program as in-line tokens. View the Pascal source code as the translator works. No file size limit; produces an approximately equal sized C file.

Includes meaningful, well-documented sample applications, and a manual that helps you to complete the translation. Supports Lattice, Desmet, and C86.

MSDOS \$130

FAST, Easy-to-Use Graphics, Royalty-Free: Essential Graphics

Draw fast dots, lines, circles, arcs, rectangles, and box fills. Draw a bar or exploded pie chart or a shaded line graph with one function call. Use the font and clip-art manipulation routines with the 10 fonts included (up to 8 simultaneously), or choose from over 500 other fonts and clip-art sets available.

Essential Graphics provides fast animation and graphic windowing, using GET and PUT, and generates compact code. Demonstration programs and comprehensive manual included.

Supports IBM Color, EGA, and Hercules cards, Epson and Oki printers. Lattice, Aztec, C86, Desmet, MS C, others. No royalties.

PCDOS \$219

C DYNAMO! WINDOWING: Full C Source, No Royalties POWER WINDOWS AND C FUNCTION LIBRARY

Power Windows covers all the bases: overlays, borders, 1-2-3 style or pop-up menus/help windows, zap instantly on/off screen, status lines, horizontal/vertical scrolling, color control or highlighting, word-wrap, files to windows, keyboard to windows. Powerful, easy to use, integrated error messages, thorough documentation. Supports IBM monochrome or color.

MSDOS. Only \$119.

C Function Library - includes 325 fundamental functions with readable source and thorough documentation.

MSDOS. Only \$119.

No matter what you have, you need these. Best value available. Highly recommended!

Clean, Thorough Debugging PERISCOPE I, II or IIX

Always available, start debugging after a crash. Source lines, line numbers and symbol support help save hours of frustrating work.

New version 2.1 enhancements include DOSEDIT-like command editing, definition of up to 4 DATA windows, increased 'monitor' breakpoint speed, and much more. Other features include: disassembly, customization by YOU, in-line assembly, full 8087/287, 286, 75+ breakpoints, EGA, and traceback. It also has source support for Lattice, MSC, C86, and MS Pascal, and symbol support for Desmet and Aztec C. Even debug drivers and resident programs.

Periscope I includes an independent, memory-protected board and break-out switch; Periscope II has a break-out switch and software, Periscope IIX is software only.

PCDOS, I \$269

II \$115 through 8/31/86

IIX \$95 through 8/31/86

First Database Exclusively for C is also Royalty-Free db _____ VISTA from . . . RAIMA Corporation

— "If you are looking for a sophisticated C programmer's database, this is it." —

— Dave Schmitt, President, Lattice, Inc.

Designed exclusively for C, db _____ VISTA is a royalty-free programmer's DBMS. Take full advantage of C through ease of use, portability, and efficiency. You optimize for speed and efficient disk storage.

Multiple key records, fast B-tree indexing, virtual memory disk accessing. Tailor db _____ VISTA to your needs by using only those features you require. ASCII file transfer utilities make moving to db _____ VISTA a snap.

MSDOS, Unix, Xenix, Macintosh, Amiga.

Single user Source \$459

Object \$179. Multiuser Source \$929. Object \$450

Even for Small Files: Convenient, Fast Access CBTREE

Why spend time writing file management code when you can use consistent, flexible, documented, professional functions? Even multiuser record locking and variable-length records are supported.

Add, delete, and update without needing to reindex. Store keys and record locations in B+ trees.

You can access any record or group of records by the value of a user specific key. Search your files from any point, forward or backward.

Full, balanced B-tree support includes use of multiple keys, unlimited number and length of keys.

Use this powerful ISAM, even if you've previously done without.

Learn how to write systems for managing large files by using CBTREE source as a guide. Modify it and transfer it to another operating environment without royalties.

MSDOS \$99

Flexible Screen Development with SECURITY CHECKING and HELP SCREENS: ZVIEW Screen Library

Use this field-sensitive tool to develop data entry screens and windows and provide run-time flexibility. Security level settings restrict inquiry or update of fields; multiple screen help display is available at screen and field level. You can also customize ZVIEW's operation and make any field characteristic change during execution.

ZVIEW gives you full control of attributes, colors, boxes, protected fields, scrolling, and more. Load screens from memory for fast response. Field support includes alpha, numeric, or alphanumeric data types, case conversion, range checking, and field comparison, and ZVIEW provides automatic data conversion to and from ASCII screen format. For Microsoft C, Lattice 3.0, and Aztec 3.2e. Supports EGA, color, and monochrome displays.

PCDOS \$219

Fastest Interactive C Development on Earth: Instant C version 2.0

Instant C's NEW version 2.0 gives you immediate (2 secs.) compilation and execution of large (5,000+ line) programs, and the ability to link in external (commercial or your own) libraries in an interactive, Lattice 3.0 or Microsoft 3.0 compatible, interpreter-like environment with an integrated full screen editor and source level debugger.

You'll get full K&R standard C, fast (33 second sieve) execution speed, and debugging with source code animation, single-stepping, backtracing, and unlimited conditional breakpoints.

Instant C now supports multiple screens and graphic devices, run-time checking of pointer and array references, and includes a new manual, expanded tutorial and reference section, and complete library source.

MSDOS \$399

**Rational
Systems, Inc.**

Call for a catalog, literature, advice and service you can trust

HOURS

8:30 AM - 8:00 PM EST.

800-421-8006

THE PROGRAMMER'S SHOP™

128-D Rockland Street, Hanover, MA 02339

Mass: 800-442-8070 or 617-826-7531 6/86

"We at Sunspot are thrilled to know that there is a store that can cut through all the "bull", and find us the products that most computer stores know nothing about. Keep up the good work."

Arland Hensler
Sunspot

Our guess is that the cause is a problem with common subexpression elimination combined with *float/double* conversion.

Overall, this is a very good compiler for a very good price. The only reason we do not recommend it is that we think you'd rather spend the few extra dollars for the developer's version. The extra goodies make the total bang for the buck impossible to beat.

Datalight C Developer's Kit

This version of Datalight's compiler, new since last year, is called the Datalight C Developer's Kit. It provides support for large-model programming (four models in all). The comments for the personal version apply equally to the developer's version, so we won't repeat them here. The developer's package is indistinguishable from the personal version except that it includes two extra disks containing the large-memory-model library files and source code for the library.

We did notice two problems with the documentation when using the developer's version. First (and trivial), the file names for the various large-model start-up object files were listed incorrectly (cc???.obj instead of c???.obj). Second (and more irritating), we could not find any mention of how to run the compiler in any of the large-model modes. We found the information in the usage summary the driver program prints out when it is run with no arguments.

Because it is the same compiler, it is no surprise that the developer's version contained the same bug we found in the personal version.

Reviewing this compiler was quite a surprise for us. For such a low price, we were expecting a "light-weight" compiler. What we got was a package that is as good as or better than most of the "heavyweights." Datalight C implements a complete C language. It also compiles quickly, doesn't take up much disk space, and looks impressive in the benchmarks. About the only things we could ask for are a source-level debugger and a more extensive library.

Eco-C88

Ecosoft started out marketing a C compiler for Z80-based CP/M machines. A little over a year ago, it started selling an 8086 compiler for MS-DOS systems at a significantly lower price. It is now up to Release 3.0 of that compiler, and it also offers a low-cost program editor to go with it. Source for the run-time library is also available at extra cost.

The Eco-C88 compiler does not allow for programming in large-memory models, but it does include a sensing library that can automatically detect and use an 8087. Bit fields are not supported, but *void*, *enum*, structure assignment, structure pass/return, and function prototyping are all supported.

Eco-C88 comes on two disks plus an optional third containing CED, the program editor. The disks and the manual come in shrink-wrap, with

no license visible on the package. An install program automates installing the package on hard- or floppy-disk systems. A cc driver program with source code is included.

The documentation is poor as far as packaging and technical information goes. There is no detailed list of what is provided, not even in the read.me file. The manual lists a *-r* option for cc as "Assemble with 8087 flag on," with no explanation of what that might mean. There are no options to control optimization strategies. Nested comments are allowed unless a *-nn* option is supplied. A *-pn* option sets the "pickiness" level of the compiler, which can lead to warnings for a large number of common usage errors (à la the Unix lint utility).

The manual is an IBM-size, 172-page paperback book and includes sections on the optional CED editor. It does not attempt to be a teaching tool. In fact, Ecosoft recommends (and will sell you) Purdum's *C Programming Guide*, and the manual sometimes refers you to that book. The section covering the library needs to be expanded. The descriptions of each function are too brief, and there are almost no examples.

Ecosoft claims the current compiler has no known bugs. It promises a free compiler to any registered user who sends the company written documentation of an actual bug.

IBM C

IBM has finally released a C compiler for its Personal Computer. Like much IBM PC software, IBM did not write

Benchmark	Aztec C	C86	Data- light C	Data- light Kit	DeSmet C	Eco- C88	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Micro- soft C	Mix C	Tool- works C	White- smiths C	Wizard C
AVERAGE																	
COMPILE	5.7	8.7	4.3	4.1	3.1	5.9	8.9	8.1	5.9	4.9	4.8	15.0	8.6	11.4	6.0	10.8	6.5
LINK	3.5	10.1	6.0	6.0	3.5	11.5	7.7	5.1	5.6	9.4	8.8	14.3	5.7	15.0	8.4	14.8	6.8
doc1																	
Min_xt	3.0	6.0	3.0	3.0	2.0	3.0	3.0	5.0	3.0	2.0	3.0	12.0	4.0	4.0	3.0	5.0	4.0
doc10																	
Min_xt	4.0	7.0	3.0	4.0	3.0	4.0	5.0	5.0	4.0	3.0	4.0	12.0	6.0	5.0	4.0	7.0	5.0
doc100																	
Min_xt	13.0	15.0	6.0	6.0	5.0	12.0	22.0	15.0	13.0	11.0	11.0	21.0	16.0	18.0	13.0	25.0	10.0
doc1000																	
Min_clt	54.0	68.0	32.0	32.0	19.0	67.0	90.0	87.0	—	56.0	58.0	61.0	91.0	66.0	111.0	40.0	34.0
Min_cs	64.0	68.0	31.0	31.0	19.0	67.0	151.0	88.0	—	56.0	57.0	63.0	90.0	85.0	110.0	40.0	46.0
Min_xt	70.0	68.0	32.0	32.0	19.0	67.0	157.0	88.0	—	56.0	57.0	62.0	91.0	127.0	111.0	40.0	46.0
No_opts	69.0	68.0	32.0	32.0	19.0	67.0	91.0	88.0	78.0	57.0	57.0	64.0	91.0	206.0	111.0	40.0	38.0

Table 8: Compile and link time

Program faster, debug faster, in C, Pascal, BASIC, dBASE*

Now, use SOURCE PRINT to clarify and organize your source listings.
You'll program more efficiently with this integrated source formatting tool:

Cross-reference Listing. SOURCE PRINT's cross-reference list shows you relationships in your program by listing the line and page number of every occurrence of each variable, function and procedure. For each of these objects, SOURCE PRINT also lists all procedures in which the object is used.

Automatic Indentation. SOURCE PRINT automatically indents source code based on structure nesting. This saves you time and keeps your source codes uniform. Alternatively, you can keep your source left-justified for speed and let SOURCE PRINT auto indent the listing.

Structure Outlining.

SOURCE PRINT automatically draws lines around nested structures on your source listing printouts, and on your monitor. Your program structures become clearer.

Nesting-error Detection. SOURCE PRINT flags structure nesting errors.

SOURCE PRINT also lets you print key words in boldface and display them emphasized or in color on your monitor, extract functions and procedures by name, generate page headings and a table of contents.

SOURCE PRINT is easy to use. All combinations of features are simply specified on the command line.

For IBM PC, XT, AT, compatibles. DOS 2 and later. ≥192K RAM. Not copy protected. Execute from floppy or hard drive.

*SOURCE PRINT works with dBASE II, III, III+ and Modula-2.

Order by mail or phone

800-257-5774 (within CA) 800-257-5773 (outside CA)

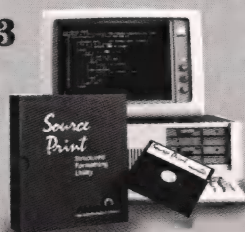
or see your local dealer

Call for free brochure

AldebaranTM
Laboratories Inc.

Developers of high-speed professional software

3738 Mt. Diablo Blvd., Lafayette, CA 94549 (415) 283-7084



C

```
05-31-86 14:27:38 demo.c
Sat 05-31-86 14:27:59

1  dobold (iar)
2  int iar;
3
4  {
5  char c, d, *p;
6  int i, j, good;
7
8  iar--; c = linebuf[i];
9  good = 0;
10
11  while (iar < nres)
12  {
13  {
14  p = ares + i; j = iline;
15  while (*p > 32)
16  {
17  p++; j++;
18  }
19  if (*p EQ 0 && j < 30)
20  {
21  good = 1; iar = nres;
22  }
23  }
24  iar++;
25  }
26
27  return (1);
28 }
29
30 err (m)
31 char *m;
32 {
33 printf ("Error
34
35 }
```

BASIC

```
06-01-86 13:22:24 simod.bas
Sat 05-31-86 14:50:14

20  FLAG=1
30  WHILE FLAG
40  FLAG=0
50  FOR I=1 TO J-1
60  IF T(I)>T(I+1) THEN
    SWAP T(I),T(I+1)
    FLAG=1
70
80  WEND
```

Pascal

```
05-31-86 14:40:02 demo.pas
Sat 05-31-86 14:40:27

1  procedure sorterr;
2  begin
3  a := 1; b := 1000; x := 2;
4  repeat
5  if b > a then
6  begin
7  setflags;
8  errprc(a, b, x)
9  end;
10 until x=0
11 end;
```

Source PrintTM

Comprehensive Formatting Tool

\$139⁰⁰ Version 1.2

License for use on a single computer.

Add \$5.00 shipping/handling. Within CA add sales tax.
MC, Visa, AmEx accepted. Immediate delivery.

SOURCE PRINT is a trademark of Aldebaran Labs, Inc. dBASE is a trademark of Ashton-Tate.

C COMPILERS

(continued from page 46)

the compiler in-house. Instead, it licensed a version of Microsoft C and repackaged it under its own label. The IBM C compiler is Microsoft C, Version 3.0, with support for a new memory model—huge—which allows a single declared object to be larger than 64K. It also has a better library.

As those of you who read last year's review already know, this is an excellent compiler. It directly supports four memory models and has language support for mixed-model programming (for example, you can dynamically allocate a 200K object and manipulate it via explicitly declared huge pointers in an otherwise small/

small program). It has five floating-point options, including in-line 8087 code with a sensing library and a faster but less accurate software-only library. There is an option to generate code for an 80186 or 80286 and all kinds of options to control optimization strategies. It also has a symbolic (not source-level) debugger and a make utility. No source code is available for the run-time library, however.

The IBM package is professional. The compiler comes on four disks. The manual covering compiler installation and use is complete.

The documentation is top-notch in almost every category. Three manuals are provided in two IBM-size (surprise!), D-ring binders with slipcases. One binder contains the li-

brary reference manual, and the other contains a manual on using the package and a smaller, language reference manual. The manuals document IBM C in great detail—the only things missing are a hand-holding tutorial on C and information about where IBM C differs from the K & R standard. Extensions include *void*, *enum*, structure and union pass/return, and function prototyping.

The language reference is complete and readable, though it is a reference document and not a tutorial. The library reference is a joy—one function per page, alphabetical order, cross-reference, large table of contents and index, good use of large typefaces, boldface and italic print—the works. The production values (paper stock, ink, and so on) are also

Benchmark	Aztec C	C86	Data-light C	Data-light Kit	DeSmet C	Eco-C88	Hot C	IBM C	Lattice C	C Prog. Sys.	Let's C	High C	Micro-soft C	Mix C	Tool-works C	White-smiths C	Wizard C
AVERAGE																	
Float_1	9693	11780	—	—	12202	—	—	17240	13242	11962	13310	—	18528	7578	8427	16165	—
Float_2	10645	—	13575	13575	—	12387	11179	21598	13542	—	—	21869	23110	—	—	—	10863
Float_3	10661	—	13585	13585	—	—	11187	21609	13550	—	—	21874	23120	—	—	—	10873
Float_4	8120	—	—	—	9813	—	6827	14585	12327	—	—	16720	15968	—	14192	14461	—
Float_5	—	—	—	—	—	—	—	21497	12830	—	—	—	23011	—	—	—	—
Float_6	—	—	—	—	—	—	—	21505	12833	—	—	—	23014	—	—	—	—
Float_7	—	10222	—	—	—	—	—	14481	12116	9447	—	—	15862	—	—	—	10985
MemMdl_2	5948	10712	9825	9825	11289	10932	4497	8006	9528	8923	8607	22280	9170	8030	12242	14933	9765
MemMdl_3	5900	—	—	11801	—	—	—	7920	9548	—	—	25484	9349	—	—	16313	10445
MemMdl_4	7292	—	—	12707	—	—	—	10758	—	—	—	28617	—	—	—	24598	13019
MemMdl_5	7478	13625	—	12870	11826	—	—	—	11531	13640	—	30240	—	—	—	26800	14204
MemMdl_6	—	—	—	—	—	—	—	—	—	—	—	—	11511	—	—	—	—
MemMdl_7	—	—	—	—	—	—	—	10959	—	—	—	33296	12286	—	—	—	18650
MemMdl_8	—	—	—	—	—	—	—	—	10857	—	—	—	—	—	—	—	13070
MemMdl_9	—	—	—	—	—	—	—	—	11643	—	—	—	—	—	—	—	14569
MemMdl_11	—	—	—	—	—	—	—	11369	—	—	—	—	12680	—	—	—	19169
minflo																	
Min_cit	4080	5520	4832	4832	4608	5266	2726	5886	7156	4786	5673	7392	6806	3213	7790	8112	3412
Min_cs	4080	5504	4832	4832	4608	5266	2694	5886	7156	4786	5673	7392	6806	3182	7790	8112	3396
Min_xt	4048	5520	4832	4832	4608	5266	2694	5886	7156	4786	5673	9472	6806	3495	7790	0	3412
No_opts	4032	5520	4832	4832	4608	5266	2726	5886	7230	4786	5673	9472	6806	3213	7790	8112	3412
minmain																	
Min_cit	2016	4306	2800	2800	1536	1600	436	2026	2370	3744	4485	4800	2480	2927	5812	1808	1674
Min_cs	2016	4306	2800	2800	1536	1600	436	2026	2370	3744	4485	4800	2480	2925	5812	1808	1674
Min_xt	1968	4306	2800	2800	1536	1600	436	2026	2370	3744	4485	6896	2480	3040	5812	0	1674
No_opts	1968	4306	2800	2800	1536	1600	436	2026	2370	3744	4485	6896	2480	2927	5812	1808	1674
minprtf																	
Min_cit	4528	9340	6784	6784	7168	8880	2818	5900	6794	5888	6873	8272	7180	4535	5834	12704	7518
Min_cs	4528	9340	6784	6784	7168	8880	2802	5900	6794	5888	6873	8272	7180	4533	5834	12704	7518
Min_xt	4480	9340	6784	6784	7168	8880	2802	5900	6794	5888	6873	18832	7180	4661	5834	0	7518
No_opts	4480	9340	6784	6784	7168	8880	2818	5900	6794	5888	6873	18832	7180	4535	5834	12704	7518
minputs																	
Min_cit	3296	4430	4224	4224	1536	2784	1400	4414	4740	3840	4584	5056	5430	3068	5926	5712	1862
Min_cs	3296	4430	4224	4224	1536	2784	1384	4414	4740	3840	4584	5056	5430	3066	5926	5712	1862
Min_xt	3248	4430	4224	4224	1536	2784	1384	4414	4740	3840	4584	7152	5430	3194	5926	0	1862
No_opts	3248	4430	4224	4224	1536	2784	1400	4430	4740	3840	4584	7152	5430	3068	5926	5712	1862

Table 9: EXE size

excellent throughout.

IBM C is very good, but Microsoft has not been sitting still.

Lattice C

Lattice was the first company to release a quality C compiler for MS-DOS systems, and it quickly established market leadership and a reputation as the compiler of choice. In last year's review, we found that Lattice C had not kept pace with its competitors and no longer deserved its billing as the compiler to beat. Lattice has since released a new, upgraded version of its compiler and has made significant improvements.

Lattice C, Version 3.0, supports a total of six memory models and four floating-point-support options. It can generate code for 80186 and 80286 processors. The company also offers a source-level debugger called C-Sprite as a separate product.

The compiler comes on four disks and includes batch files for hard-disk and floppy-disk installation, though you may decide to delete some of the subdirectories created on a hard disk if you don't need all memory models available at all times. Using the compiler is complicated because usage information is in three places: the basic manual, which applies to Version 2.15; the Version 3.0 *Technical Bulletin*; and the read.me file. Lattice has added, removed, or changed numerous options, so you have to read everything carefully to build up a picture of current reality.

We had a serious problem regarding the linker. The *Technical Bulletin* warns that MS-DOS LINK, Version 3.0 or later, may be required in certain cases. Careful reading led us to believe that the warning did not apply to what we would be doing, and then the read.me file claimed the problem was resolved in any case—LINK, Version 2.0 or later, should work in all cases. We therefore tried to use the version of LINK (2.2) that came with our computer, as we did with all compilers that did not supply their own linker. It did not work, no matter what memory model we tried. The usual symptom was "write fault error writing device PRN" followed by a system crash. We were not redirecting output. LINK 3.01 worked fine, so that is what we used, but it gave Lattice an unfair advantage as

3.01 is faster.

An environment variable (*INCLUDE*) is supported to specify a search path for include files. In fact, using it is mandatory if you use angle brackets to delimit the include file name, as is commonly done for *stdio.h*. Lattice C does not search the current directory, or even directories specified with the *-i* option, automatically—no *INCLUDE*ee, no *findee*.

An option is available to force word alignment for all data types except *char* and *struct*. We used this op-

tion because non-word-aligned fetches are less efficient on 8086 (or better) machines. We think word alignment should be the default.

An option has been added to cause the intermediate (quad) file in memory to speed up compile time. We used this option for all but *doc1000.c*—apparently we ran out of memory after 600–650 lines, resulting in an "intermediate file error."

The documentation consists of two IBM-size, spiral-bound manuals. This is a change from last year, when Lat-

C Programmers: If you do business application development, you need

FAST PROGRAMMING™

The C programming language tool for business applications.

Why Fast Programming?

Because it is a **complete** development system. You will no longer need to integrate incompatible libraries and fragments of programs.

Because finished, ready-to-compile C programs are generated.

Because there are **no run time royalties**. Not even for the utilities.

Because the resulting system is truly multiuser.

Fast Programming is \$995 for a site license. VISA, MC, AMEX accepted. Currently available versions include PC-DOS, XENIX and UNIX. Call for specific machine availability.

Subject, Wills & Company

800 Enterprise Drive
Oak Brook, Illinois 60521
(312) 789-0240

The components of Fast Programming are:

C ROUTINES

Decimal math, keyin and display with windowing, extended string handling, time/date conversions, time/date edits, misc edits, sequential I/O, key sequential I/O, random I/O, indexed I/O, printer output, error handler, system interface routines and more.

C PROGRAM GENERATOR

Maintenance and data entry program generator, report program generator, processing program generator, I/O routines generator. The generators provide many user exits in the generated source code. This allows for custom coding without changing the generated C source program. This means that no custom coding is lost when programs are regenerated.

RUN TIME UTILITIES

SORT, INDEX, REFORMAT, BUILD, CREATE, DUMP, CHAIN, ENCODE, FILECHK, LIST, RENAME, COPY and DELETE.

MENU SYSTEM

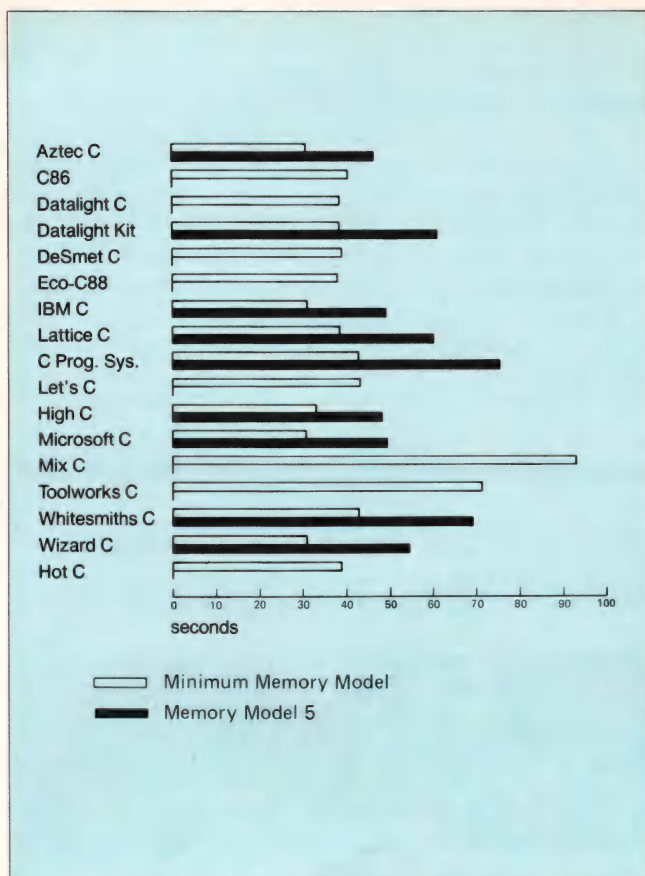


Figure 1: Execution times of pointer benchmark

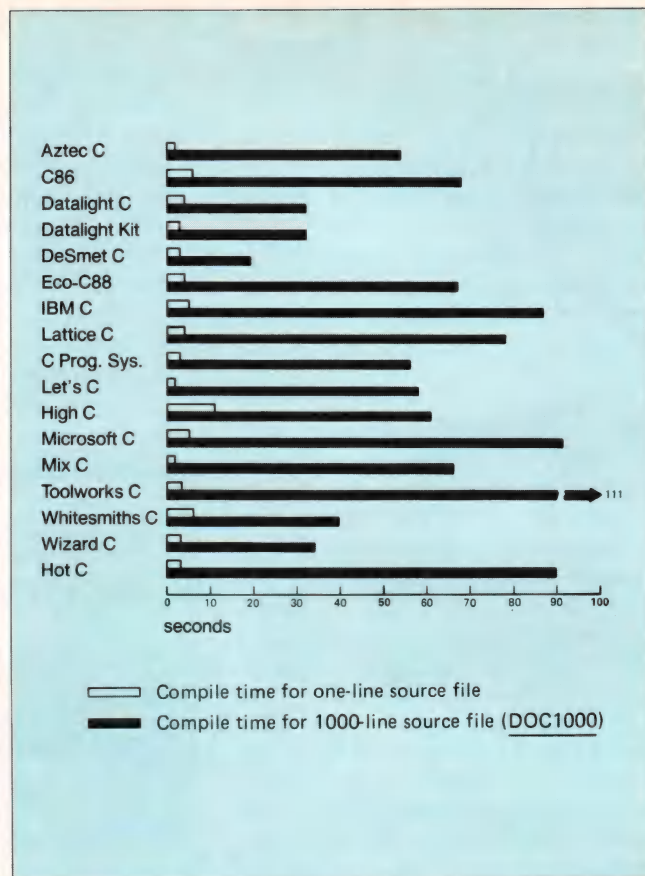


Figure 3: Compile initialization overhead

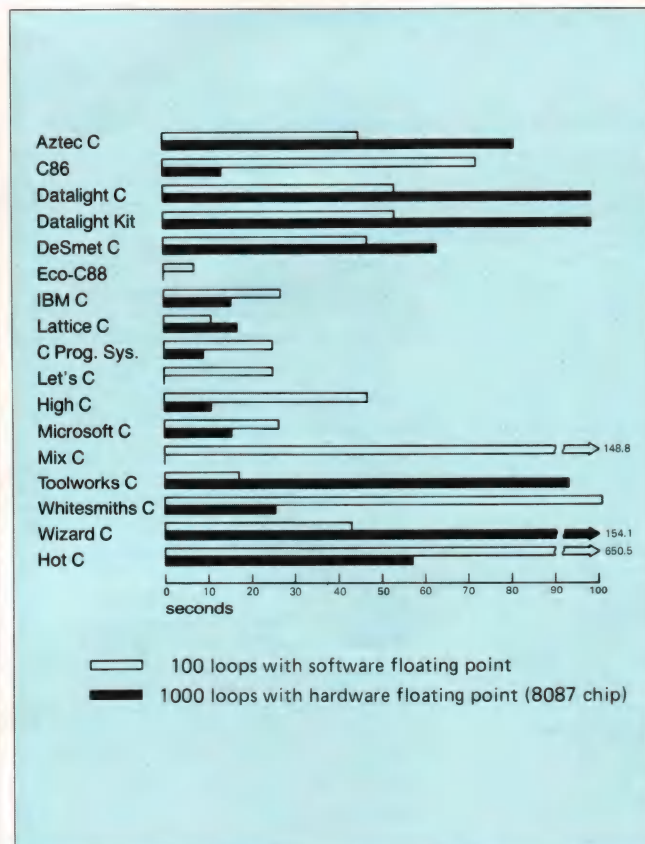


Figure 2: Execution times of trig benchmark

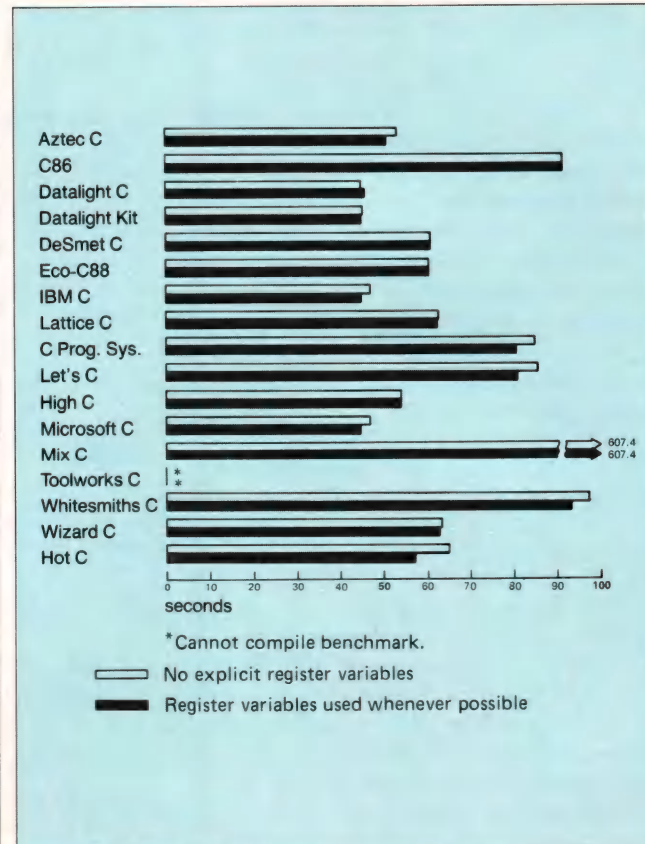


Figure 4: Use of register variables with dhrystone

tice supplied the normal D-ring binder and slipcase. At least one of our reviewers prefers the spiral-bound approach because it takes less space and lies flat, but update pages cannot be inserted.

The first manual is the reference for Lattice C, Version 2.15, and the second (almost as big as the first) is a technical bulletin describing the changes for Version 3.0. The production quality (use of typefaces, color, and so on) is excellent in the first manual, less so in the second. Also, the library reference is split between the two. You start with the *Technical Bulletin*, which may point you back at the 2.15 manual. The 2.15 manual includes a table of contents, index, and a separate index for library functions. The *Technical Bulletin* has a brief table of contents and an updated function index.

The 2.15 manual contains a brief summary of the language definition as well as a list of differences from K & R; it's well done but will not take the place of a language reference manual. The library reference is in the Unix style, with several functions sometimes described together and no examples, but the amount of information is good and the function index makes lookup easy (except for needing the two manuals). Cross-references and warnings are included where appropriate, as well as an indication of how portable you can expect the function to be. Overall, we would rate the documentation very highly if the current two manuals were replaced with a single volume specifically for the new release and up to the standard set by the 2.15 manual for production values.

Although Version 3.0 represents a significant improvement for Lattice, the other vendors have not stood still either and we still cannot rate this compiler in the top few. It should be noted, however, that Lattice provides compilers for the same language for minicomputers and mainframes as well as for many other microcomputers. Also, Lattice still enjoys more third-party support for add-on libraries and utilities than any other vendor, although that gap is narrowing. These considerations will probably

be important for some users.

Manx Aztec C

Manx Software Systems got its start selling a very good 8080 compiler for CP/M systems at a reasonable price. It has since moved that compiler to MS-DOS machines, Apple IIs, the Macintosh, and most recently the Commodore Amiga and the Atari ST machines. It has made many enhancements and upgrades along the way, but the same basic language is available on all those machines from

the same vendor. For MS-DOS machines, the company markets several different packages with different bundles of goodies. We reviewed the Commercial package, which comes with everything, including full source code for the run-time library. We reviewed Version 3.30C of the compiler while it was still in beta test.

This is a big package. The latest release includes a new source-level debugger and an execution-time profiler. Also included are a full-screen editor similar to the Unix vi editor, a

C-terp

The C Interpreter You Won't Outgrow



C-terp will grow with you as you progress from novice through professional to guru. Unbelievable, but true, the easiest-to-use C interpreter will provide you with the most advanced programming features for upward growth. Our exclusive **object module support** enables you to add libraries (like HALO, PANEL, Windows for C, etc., or your own homebrew libraries) to C-terp as you add them to your computing repertoire. Use C-terp as a microscope on your libraries! Flip a bit and allow our **software paging (NEW)** to handle those big jobs! There are no fixed-size tables to overflow, and C-terp can be configured for different screens and screen adapters (NEW). With multiple modules and **full K&R support**, we offer a dream C environment.

- Our new improved **configurable editor** competes with anything going.
- Speed -- Linking and semi-compilation are breathtakingly fast.
- Convenience -- Errors direct you back to the editor with the cursor set to the trouble spot.
- Symbolic Debugging -- Set breakpoints, single-step, and directly execute C expressions.
- Compatibility guaranteed -- batch file to link in your compiler's entire library. Supported compilers include: **Computer Innovations C86, Lattice C, Microsoft C 3.0, Mark Williams C86, and Aztec C.**
- Many more features including batch mode and 8087 support.

What Our Users/ Reviewers Are Saying

- "... easy to use, powerful, and a timesaver."
- "... we absolutely LOVE C-terp."
- "... has restored my faith in interpreters."
- "... a programmer's dream."
- "... wonderful technical assistance."
- "... increased our productivity by a factor of 40."
- "... the best C product ever, in any category."

- **Price: \$300.00 (Demo \$45.00)**
MC, VISA

Prices include documentation and shipping within U.S. PA residents add 6% sales tax. Specify compiler.

- C-terp runs on the IBM PC (or any BIOS compatible machine) under DOS 2.x and up with a suggested minimum of 256 Kb of memory. It can use all the memory available.
- * C-terp is a trademark of Gimpel Software.

GIMPEL SOFTWARE

3207 Hogarth Lane • Collegeville, PA 19426
(215) 584-4261

fairly powerful make utility, extra libraries for MS-DOS screen and graphics functions, a CP/M-86 library, and other utility programs familiar to Unix fans. The compiler supports four memory models and four floating-point-support options and can generate 80186 or 80286 code. It supports the full K & R language plus *void*, *enum*, and structure assignment.

The Commercial version is distributed on four disks, but the compiler, assembler, linker, and libraries for all memory models will fit on two disks with room left over. Aztec C uses its own object format, but a utility is provided to convert to Microsoft format if desired. The installation notes concentrate on floppy-disk installation and there is no batch file for automating the process, but installation is not too difficult.

Aztec C supports environment variables specifying the search path for include files and libraries. If no *INCLUDE* environment variable is set, the compiler scans the current directory for include files enclosed in angle brackets, which is nice of it. Manx includes an older, nonoptimizing version of the compiler, which it claims will compile faster than the full optimizing version.

New with this version is a driver program that can handle wildcards on the command line and can run the linker automatically. Previously, the compiler would invoke the assembler, but you had to run the linker separately.

The documentation has been expanded and improved since our last review, and it now has an index. The manual consists of more than 630 pages in an IBM-size, D-ring binder with slipcase. It is organized as named sections that desperately need index tabs for quick flipping. The overall table of contents at the front covers all sections in detail except the various library sections. Each section has its own table of contents as well. The index is fairly complete but would be easier to read if it were better formatted. Page references are to "section name.page number." This is where the index tabs would come in handy—it's not easy to find your 40-page section in

the middle of 600-odd other pages. The index starts off with a list of how the sections are ordered, but that is not much help (especially if you have reshuffled them to put all the library sections together).

There is a lot of material here describing the compiler, assembler, and linker and technical information such as memory layout, building overlay programs, and ROMable code. An excellent section (25 pages) on compiler error messages has a paragraph describing each error, often with examples!

No language reference is included; you will need K & R or some other book. A good section describes implementation-specific information and differences from K & R. There is also a section called "style," which contains some good advice on C philosophy and programming practices. This is an unusual touch.

The library sections are organized à la Unix, with several related functions on a page. The function descriptions are typically terse, but there are sometimes examples. The typefaces and use of boldface, italics, and indentation are good. The new index is the easiest way to find a function you know the name of; otherwise it may take you a while.

C Programming System

Mark Williams Co. has been around for quite a while and has established a good reputation for its Unix-like operating system and its C compiler. The C Programming System compiler is in the "heavyweight" class and is priced accordingly. It includes several 8087 support options and large/large-memory-model support. It comes with a full source-level debugger, two editors (including full source code for MicroEMACS—an EMACS subset), and several utility programs. CSD, the source-level debugger, is powerful but can be used only with the Mark Williams object format, which in turn can be used only for small/small-model programming.

C Programming System comes with a total of five disks and two IBM-size manuals in the usual D-ring binders with slipcases. The manual section on installing the compiler applies to the company's Let's C, which is a little disconcerting. The release notes in front of the manual

cover installation of C Programming System. Although a list of files on the distribution disks is given, the list does not describe what each file is.

A large number of compiler options are available to turn on (or off) various categories of warning messages, which can prove useful. There are no options for controlling compiler optimization strategies as there are in most heavyweight compilers. There is an option that causes a different version of the start-up and exit code to be linked in that can produce smaller .exe files if you don't need the standard I/O package. The compiler also has options that cause it to generate code specifically for 80186 and 80286 processors.

The documentation is professional and includes a good table of contents and an index. The language reference material is simply a list of differences from K & R. The library reference is complete but perhaps a little too concise and does not include examples.

The manuals do have some confusing inconsistencies. The release notes, for instance, indicate that MASM (the Microsoft assembler) should be on your program disk if you want to use Microsoft object format, but it is not clear why. We did not need MASM to compile and run any of the benchmarks. The section on the assembler claims that the large model will be discussed, but it isn't. The assembler option to generate Microsoft object format is described as causing small-model object to be generated (!).

Mark Williams' C Programming System has little to recommend it over its competitors and its debugger is no longer unique.

Let's C with CSD

Mark Williams' Let's C package, new since last year, is a stripped-down version of the company's established C Programming System. Let's C includes the same compiler but with support for Microsoft object format and 8087 floating-point coprocessors removed. Because you are restricted to Mark Williams object format, you can program only in the small/small-memory model with Let's C. Also removed from the Let's C package are the make utility, ed editor, and m4 macro processor. Not removed from

How to tackle a 300 page monster.

Turn your PC into a typesetter.

If you're writing a long, serious document on your IBM PC, you want it to look professional. Precise. Easy to read. You want MicroT_EX.

MicroT_EX was designed especially for desktop publishers who require heavy duty typesetting. It is based on the T_EX standard, with tens of thousands of users worldwide. Documents from smaller than 30 pages to 5000 pages or more. And that's something that other programs just can't match.

No other PC software gives you as many advanced capabilities as MicroT_EX. Superior hyphenation control, the sophistication of ligatures (ffi, æ) and kerning; down-loadable fonts; aesthetic handling of math ($\pi = f'(x)$) and foreign language characters; complex table construction and multi-column tasks; dot matrix, laser printer and phototypesetter output. When used with our L^AT_EX macro package, it automatically enumerates and cross-references pages, sections, footnotes and illustrations. Plus it automatically creates your indexes, tables of contents, and even updates them for you after last minute insertions.

So if you want typesetting software that's as serious as you are about your writing, get MicroT_EX. **Call toll free 800-255-2550** to order or for more information.* Order with a 60-day money back guarantee.

MicroT_EX[™]
from Addison-Wesley

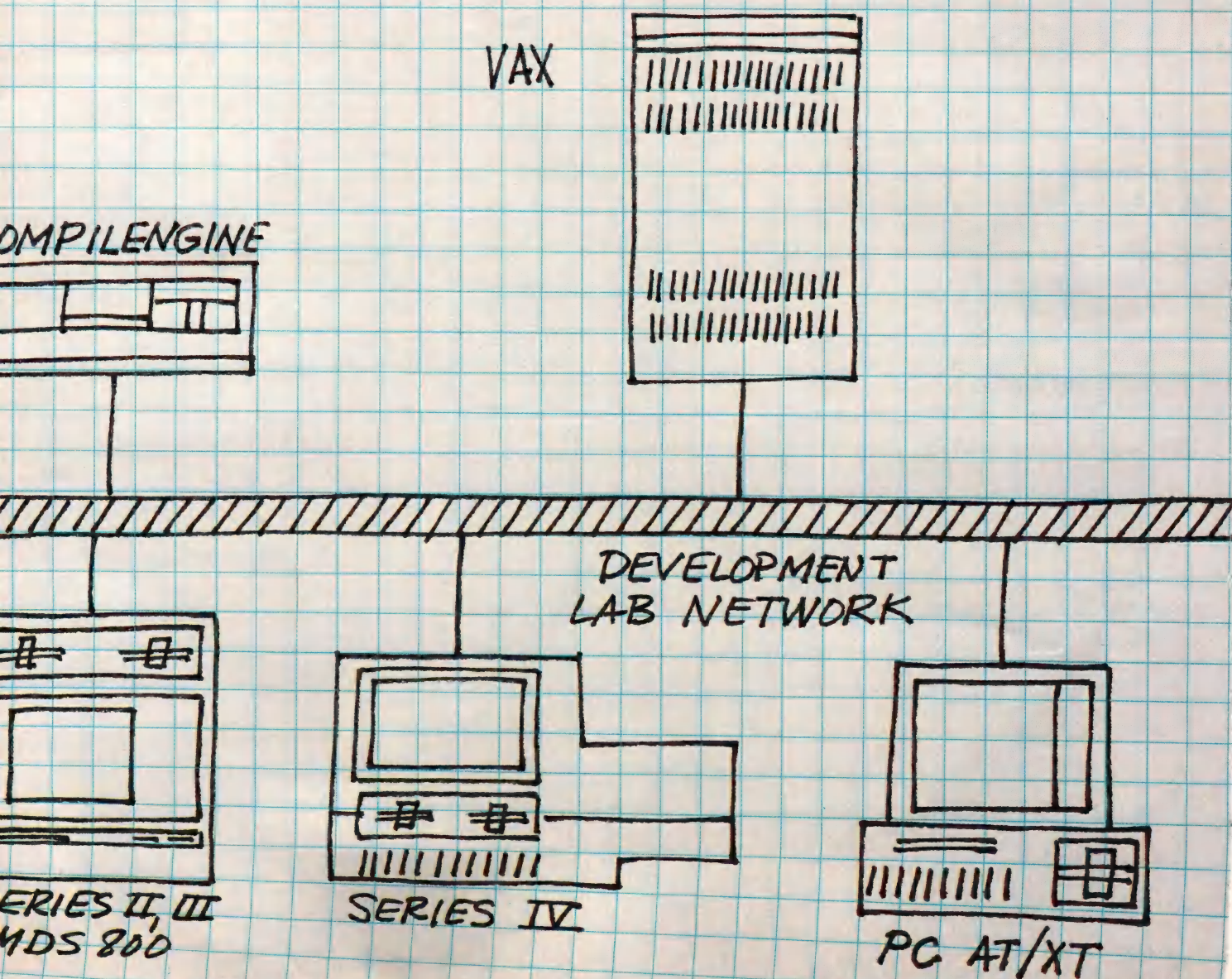
**Serious typesetting for
serious desktop publishers.**

*Dealers, call our Dealer Hot Line: 800-447-2226
(In MA, 800-446-3399), ext. 2643.

Circle no. 92 on reader service card.



NOBODY ELSE HAS MADE THE CONNECTION.



There's only one way to link the extensive resources of Intel's microprocessor development system with the power of a DEC VAX® and the affordability of the IBM® PC.

And that's via our enhanced version

of OpenNET™ for the development lab.

Thanks to this open architecture network, engineers can have immediate, and transparent, access to other team members' work.

Plus you have the ability to add special-purpose hardware to make those teams more productive.

For example, by connecting our high-performance file server, the Network Resource Manager, you can off-load file management and job distribution from the shoulders of your design team. On the off chance nobody wants to spend his time chasing down floppy disks.

And then you can hook up an 80286-based Compilengine to off-load compute-bound compilations from VAXs and workstations. Leaving them, and their human partners, more time for interactive tasks.

Equally important, the network maximizes the value of your existing development

hardware while minimizing your outlay for new equipment.

That's because OpenNET adheres very rigidly to some very flexible standards. Standards like Ethernet/IEEE 802.3. And the ISO message delivery and Intel/IBM/Microsoft® Network File Access protocols.

All of which means existing development hosts, languages and tools, including ICE, are instantly compatible with the latest ones. So you can avoid obsoleting one set of tools just to use another.

We'll even take full responsibility for servicing and supporting your network. Anywhere in the world.

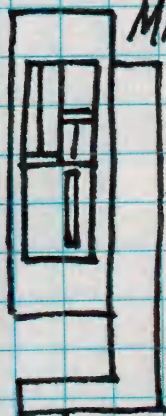
Sound like we've got things together? Then call us at (800) 548-4725.

Ask to meet with one of our experienced network engineers. We'll see you make all the right connections.

intel®

VAX is a registered trademark of Digital Equipment Corporation. IBM is a registered trademark of International Business Machines Corp. OpenNET is a trademark of Intel Corporation. Microsoft is a registered trademark of Microsoft Corporation. © 1986 Intel Corporation.

**NETWORK
RESOURCE
MANAGER**



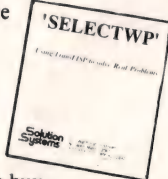
SYSTEM 310

Is LISP Right for Your Expert System?

Find Out — FREE

You can explore LISP by examining a complete sample problem. Call and we will send you a free source listing of "SELECTWP".* It prompts users for criteria and helps them choose which micro word processor to buy.

Look over the TransLISP syntax (COMMON LISP compatible). Your application will probably have similar characteristics.



Power & Flexibility

Do you get flexibility in PASCAL and C? Of course, but examine the listing of SELECTWP to see how much more power and flexibility you get. The LISP advantages:

- forward references make program flow fit the problem
- manipulate data structures of varying sizes
- create your own language to fit the problem domain
- avoid mundane, busy work required with traditional procedural languages
- powerful function and macro building facilities provide better data abstraction

*SELECTWP includes 512 lines of LISP code and 335 lines of comments.

Full refund if not satisfied in 30 days.

TransLISP gives You the Advantage

Using TransLISP for your expert system has several advantages over other AI tools. And you will see SELECTWP illustrate:

- * the ability to control how decisions are made
- * the freedom to assign weights and react to user choices
- * the complete control you have over how a problem is solved, and interaction with the user

Nothing to lose

Examine LISP carefully by studying a practical program free.

Or buy TransLISP risk free. SELECTWP is just 1 of over 20 sample programs in the complete TransLISP system. The other sample programs include: an adventure game, a program to read dBASE SDF files, "Job Counselor" and more. Use the modular tutorial, the complete 300+ function LISP interpreter, and the online help, to get started in LISP in only a few hours.

Develop programs of up to 12000 lines on a 640K system or use TransLISP on a floppy only, 256K RAM machine. MSDOS.

Call 800-821-2492 for SELECTWP FREE. Or order the complete TransLISP system risk free for only \$75.

**Solution
Systems**

335 Washington St., Norwell, MA 02061 (617) 659-1571

Circle no. 148 on reader service card.

The C Programmer's Assistant

C TOOLSET

UNIX-like Utilities for Managing C Source Code

No C Programmer should be without their assistant - C ToolSet from Solution Systems. The package consists of several utilities designed to help make C programming tasks easier.

C ToolSet (formerly C Helper) includes:

DIFF - Compares text files on a line-by-line basis or use CMP for byte-by-byte - indispensable for showing changes among versions of a program under development. So "intelligent" it stays in synch even when you add 100 lines.

GREP - Regular expression searches - ideal for finding a procedural call or a variable definition amid a large number of header and source files.

FCHART - Traces the flow of control between the large modules of a program.

PP (C Beautifier) - Formats C program files so they are easier to read.

XREF (CCREF) - Cross references variables from a program.

Available For MS-DOS - \$95

**ONLY
\$95**

Source Code Included

**Solution
Systems**

335 Washington St.
Norwell, MA 02062
617-659-1571

800-821-2492

Circle no. 152 on reader service card.

C COMPILERS

(continued from page 52)

the Let's C package is MicroEMACS, a subset of the EMACS full-screen editor with full source code. Let's C can generate the same large set of warning messages as the full C Programming System package and supports the same facilities for minimizing .exe file size.

The Mark Williams source-level debugger CSD works with Let's C, but it is an option that doubles the cost of the package. The Let's C package with the CSD option includes three disks and two manuals. Unlike C Programming System, the compiler and debugger manuals are spiral-bound, making it impossible to insert update pages. The contents of the manuals appear to be the same as the larger slipcase versions that come with C Programming System. Both use shading to mark text that does not apply to Let's C. The spiral-bound CSD manual that comes with Let's C covers only CSD, not the other utility programs that are not included with Let's C. Other than that, and the physical packaging, the Let's C documentation is the same as that for the full C Programming System package.

High C

MetaWare is new to the MS-DOS C compiler marketplace this year, but it is not without experience in providing high-quality compilers. The principals of MetaWare are well known and respected in computer language circles.

High C is large. It comes on seven disks and provides an installation batch file whose use is virtually mandatory. The compiler itself is a single .exe file of more than 520K (the compiler will run on 320K machines because much of the .exe file is overlays), so do not buy High C if you do not have lots of disk space (about 2 megabytes) left on a hard disk.

High C supports five memory models and can be directed to use a sensing 8087 library or in-line 8087 code. It supports almost the entire emerging ANSI standard, and MetaWare can change the language almost as fast as ANSI can generate new drafts. The compiler produces excellent diagnostic messages, detecting many less than desirable code features



Western Computer 286 Turbo™

Standard Features

- IBM PC/AT Compatible with 512K RAM
- Up to 1 Megabyte of system memory on the main board
- Switch selectable 8 or 10 MHz operation with one wait state on memory access (80286-10)
- One parallel port/one serial port and Clock/Calendar on main board
- Optional Serial Port on board
- Mass storage options from 20 to 140 Megabyte Hard Disk Drives and 20 to 60 Megabyte tape backup systems
- EGA video options available for CAD/CAM, word processing, etc....
- One year factory warranty

Western Computer has high quality computers at very competitive quantity pricing.

Western Computer XT Turbo™

Standard Features

- IBM PC/XT compatible with 640K RAM on main board
- 4.77 or 8 MHz operation (CPU board made in U.S.A.)
- Two 360K Floppy Disk Drives
- Hercules compatible monochrome graphics controller (720 x 350) or IBM compatible color graphics adapter (320 x 200 - four colors or 640 x 200 - two colors) and parallel printer port
- Monochrome monitor (Composite or TTL input)
- IBM PC/AT style keyboard
- Various hard disk drive and tape backup options available
- One Year Factory Warranty



Western Computer

17781 MITCHELL STREET
IRVINE, CA 92714 U.S.A.
PHONE (714) 553-1611
Customer Service Only
(714) 553-1705

TELEX 7566731 - Answer Back
WESTERN COMP
FAX (714) 553-0236

Western Computer

"For High Technology and Performance"

European Head Office

BELECTRONIC SA,
RUE CENTRALE 43
CH-1880-BEX, SWITZERLAND
PHONE (025) 631250
TELEX 456 168
Answer Back BELE CH.

IBM®, IBM PC®, IBM AT® are trademarks of International Business Machines Corporation.
Circle Reader Inquiry Number 211

without complaining about too many intended code sequences. Pragmas are supported by High C in a big way. With them you can change the segment, group, and/or class of any object; select the default calling convention; specify how values are to be returned; enable automatic register allocation or disable registers altogether; turn on or off any optimization; and generally control your compilation in any desirable way at any point in your source code. The flexibility is daunting at times, but we think we would rather make these decisions than not.

The documentation for High C consists of about 720 pages in a single IBM-size, D-ring binder. That's a lot of pages, but they are separated into three major and three minor sections by labeled index tabs. The three minor sections are the release notes, the license/warranty section, and an installation guide. The three major sections are the Programmer's Guide (usage and technical information), the Library Reference, and the Language Reference. Each of the major sections has a complete table of contents, index, and a request for user feedback. All the indexes are the permuted type found in some Unix documentation, and instructions are included on how to use one. Although some prefer this type of index (including one reviewer), some of us feel a normal index is easier to use. The index references are to section numbers, not page numbers. This can be more precise (pointing at a paragraph instead of a page), but it makes it harder to flip to the right spot.

The production quality is variable. The Language Reference is still in dot-matrix format and is hard to read in places, though the feedback request states that a typeset version is in the works. The Programmer's Guide and Library Reference are typeset, but all three sections suffer from too little white space for good readability. Also, the choice of type styles and sizes is often strange, further degrading readability. This is especially true in the Library Reference.

As far as content goes, there is an immense amount of information here, as you could guess with 720

pages and almost no white space on the average page. This is a big, complex product geared toward professional programmers who are already quite familiar with C, and such people will find all they could want in this manual. Average users will probably find it heavy going and will be in for some frustration.

The Language Reference is actually a precise, formal definition of the C language as implemented by MetaWare. It is complete and unambiguous, but if you are not familiar with formal notation and context-free grammars, you may find it difficult to understand. The Library Reference is organized along the same lines as in the emerging ANSI standard, with functions grouped according to type and types grouped by a standard header file. Within each group functions are described one at a time and in alphabetical order, but there are many groups so you will need to learn to use the index. There are examples and "cautions" and "system dependencies" sections where appropriate. In terms of content, this is excellent documentation. Overall, you are probably either going to love this manual or hate it.

Microsoft C

Microsoft is not like most vendors, which are almost continually releasing minor revisions to their products. Microsoft doesn't take many steps, but the ones it takes are big ones. Last year, just before we started working on our review, it discontinued marketing a repackaged Lattice compiler and released its own product. In one step it went from also-ran status to supplying the most professional package with perhaps the best C compiler available. For the next year it released no updates, unless you count the IBM version, which added the huge-memory model. Now, just barely in time for this year's review, comes a beta copy of the upcoming Microsoft C, Version 4.0, and it is another big step.

Compared to Version 3.0 (and IBM's), the new package features faster compile time; improved code generation; more memory models; various language and library enhancements; and a mouse-driven, full-windowing, source-level debugger that puts the rest to shame. If it

were not beyond the scope of this review, we could go on for some time about the new debugger. Suffice to say that, although the basic capabilities provided are similar to other full-function source debuggers, the user interface is a dream come true.

Like its predecessor, the Microsoft C, Version 4.0, package concentrates on the compiler itself. A powerful version of the make utility is provided, and of course that marvelous debugger, but that is about it as far as goodies go. It has no editor, no Unix tools, and no special-purpose libraries, and only source code for the start-up code is included. The compiler supports the full K & R definition of C, plus *void*, *enum*, signed, structure and union assignment, structure and union pass/return, and function prototyping (but not completely). Additional keywords are optionally supported for mixed-model programming and alternate function-calling conventions. The package has full support (compiler and library) for five memory models and partial support (compiler only) for a total of 18 memory-model variations! There are also seven floating-point-support options, including an alternate software-only library that sacrifices precision for speed.

This is a large package. It comes on seven disks, of which five disks are required to hold the compiler, linker, and all the libraries. The other two are for the start-up sources and the debugger. There are no batch files to automate the installation process, but separate sections in the *User's Guide* give step-by-step instructions for a "quick" hard-disk or floppy-disk installation. A huge amount of setup, usage, and technical information is provided. It is well written, but it takes a while to wade through. You can choose between two driver programs—MSC and CL, which is designed to work more like Unix drivers do. We preferred CL, but both support a vast number of options taking several pages to list in the summary sections provided. There are many options controlling optimization strategy, and there is an option to tell the compiler to do a simple syntax check rather than a full compile—very nice!

We received a preproduction copy of the documentation printed on

New for 'C'

A "C" programmer's tool to increase screen development productivity for the IBM PC. Security checking and help screen display are available at both the screen and field level. The automatic conversion of data types, to and from ASCII screen format, and the many other productivity-oriented features, set ZVIEW apart from the rest.

Screen Painter Highlights:

- Border colors and all character attributes and colors are supported.
- Draw single or double lined boxes using preset key strokes.
- Two field sensitivity settings to facilitate the moving and adding of fields, without destroying existing field characteristics.
- Three types of fields are available: "Protected," "Unprotected" and "Heading." The number of fields is limited to 600!
- Both 40 and 80 column screens are supported.

Optional Field Characteristics:

- Choose left or right justification, with zero or blank fill.
- Automatic key stroke conversion to upper or lower case.
- Edit fields to be numeric (signed or unsigned), decimal (zero to six decimals supported), alpha or alphanumeric.
- Display numerical values with or without commas inserted.
- All "C" data types are supported, including a special long value which is displayed as a decimal field.
- From and to range checking and character matching edit.
- Security level settings to restrict inquiring or updating of a field.
- Override ZVIEW's default tabbing sequence.
- Assignment of a single or multiple screen help file, to be displayed when the field level help key is pressed.
- Compare one field to three other fields on the current screen.

Program Interface Highlights:

- Only nine run-time library functions control all aspects of the program to screen interface.
- Dynamically change any field characteristic at run-time.
- A wide range of run-time variables to further customize ZVIEW's operation.
- One call to ZVIEW's "Waitkey" function, performs all field edits and program to end user interface.
- Automatic data conversion of all data types to and from data structures and buffers. Data goes directly from data type to screen format and back, with one call each way.
- Display screen files from disk or memory.
- Scroll function replaces vacated lines with data you provide

Windows:

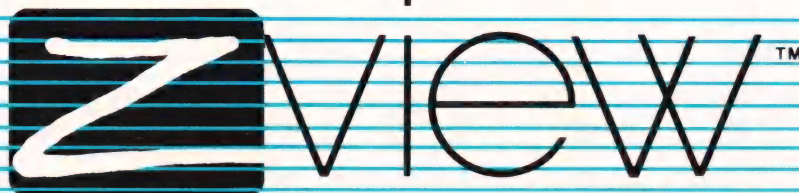
- Windows are a built in feature of ZVIEW.
- Automatic handling of the window overlay process.
- Windows are fully functional for data display and data entry.

Requirements:

- Microsoft 3.0, Lattice 3.0, and Aztec 3.2e compilers currently supported.
- IBM PC, XT, AT or compatible, MS/PC DOS, one 320k drive and a CGA, EGA or monochrome adapter.

Price:

- \$245
Includes manual and a detailed program example.



TO ORDER CALL TOLL FREE 1-800-423-0930

Customer Service and Nevada residents:
call 1-702-798-5910



Or Write: Data Management Consultants
5325 So. Valley View Blvd. Suite #7
Las Vegas, NV 89118
Master Card, Visa or company
check accepted

IBM PC, XT, AT and PC-DOS are trademarks of International Business Machines.
MICROSOFT and MS-DOS are trademarks of Microsoft.
ZVIEW is a trademark of Data Management Consultants

Circle no. 263 on reader service card.

8.5×11-inch paper, but it is a good bet that the final package will follow the pattern of Version 3.0's and now IBM's. The organization and content is an improvement in some ways over the IBM package, although it is similar except for the addition of a fourth manual documenting Code-View, the source debugger. The final package will come in three binders. One of the improvements is the addition of several appendices in the *User's Guide* with information on converting from earlier versions, writing portable code, and a summary of differences from K & R.

Assuming production values in the final package are in line with past practice at Microsoft, the documentation once again gets the highest rating. The only fault we can find is with the *User's Guide*. Although it contains a huge amount of good information, well organized and written, it has some gaps. For instance, it gives an actual example of code that will misbehave if you tell the compiler you won't do aliasing (an optimization option) and then you do—it does not give a complete list of the aliasing assumptions made by the compiler. Is it safe to point at a named array (a common practice) and access it both ways or not?

Mix C, ASM Utility, Mix Editor

Mix Software is a relative newcomer to the C compiler market. It has a very-low-cost and fairly complete product. Mix C does not generate assembler code and does not use Microsoft object format. The extra-cost ASM utility can be used to convert .OBJ files to .MIX format. Mix C can generate only .COM files, so program/data size is even more restricted than with other small/small compilers.

We received the Mix C compiler with the extra-cost ASM utility and Mix Editor, so our package included three disks and two books. Two disks and the thick (about 440 pages) book make up the compiler and ASM utility, and the third disk and 100-page manual are for the editor.

The manuals are not the usual IBM-size binders. Both are 8.5×11-inch, bound books, so updates are strictly in the form of read.me files. Techni-

cal information about the compiler is almost nonexistent, and you have to use DEBUG to change various compiler option defaults. The compiler manual is separated into five sections, each with its own table of contents. The first section, Getting Started, tells you how to set up your disks and compile and execute your first C program. The second section is a large, well-written tutorial to introduce fledgling C programmers to the language. The third section is a complete reference manual for C, and the fourth documents the supplied library functions. The last section documents the use of the C compiler, linker, and other tools provided with the package. The second, third, and fourth sections have their own indexes, but the whole book has no global table of contents or index.

Mix seems to be targeting this package at newcomers to C who will buy the book and get the compiler to boot. From that standpoint, the documentation is very good. The tutorial and reference sections are well written, with lots of examples and an appropriate level of verbiage. A few of the library functions are documented in the language reference section and can therefore be difficult to find. Mix C is probably a good choice for someone who wants to learn about C without spending much money.

Toolworks C with Mathpak

Software Toolworks has been marketing several low-cost tools and utilities, including a subset C compiler, for a long time. It has recently addressed a major shortcoming of its C compiler by making available an optional Mathpak package that adds support for the *long* and *float* data types. This is what we reviewed.

Even with the Mathpak, Toolworks C is a subset compiler. Bit fields and *typedef* are not supported, declarations are not allowed in nested blocks (only at the start of a function), double precision (*double* is a synonym for *float*) is not supported, and the preprocessor does not support *#line* or *#define* macros with arguments. Also, function calls must have exactly the same number of arguments as the function definition (*printf* and *scanf* are the only exceptions). Of these, the lack of *typedef* and parameterized macros are the most serious omis-

sions. The Mathpak does include support for an 8087, although not via a sensing library. There are some brief notes on how to modify the package to merge the software-only and 8087-only libraries to form a sensing library, but they are for hackers only—for instance, you are left on your own in figuring out how to do the actual sensing. Full source code is provided for all libraries.

The Toolworks compiler is a throwback to the old days: It comes packaged in a Ziplock bag! The compiler takes up two disks, and the optional Mathpak takes up another two. The installed package does not take much disk space, however. You have to install the compiler first and then do the Mathpak installation procedure, which modifies the installed compiler and replaces the library. It is actually easier than it sounds.

There is no driver program for the compiler. Each pass must be run separately from the command line or from a batch file. Several compiler options are available to control such things as when string constants are generated, the size of the *switch/case* table, string space, *#define* table space, and so on. Toolworks C overlaps identical strings by default, but you can override this. There is an option that causes an execution-time profile to be generated when the program is run. The option to specify an include-file search path is mandatory if you enclose the file name in angle brackets on your *#include* directive or the file will not be found. Toolworks C also has a program to configure the compiler to make a specified set of options the default.

The documentation consists of 8.5×11-inch sheets stapled together and folded to fit in that Ziplock bag. The compiler manual has 75 pages and includes a table of contents and an index, which also indexes the library functions. The 23-page Mathpak manual has a table of contents but no index (hardly necessary here). There is a very brief language summary, obviously not intended as a learning aid. Likewise, the library function reference is very brief, not even adequate in our opinion. The documentation contains an unusually large amount of technical information, indicating that this package is aimed at the technically minded.

C FEVER *catch it!*

It's becoming an epidemic... everyone is switching to C! First there were a few hackers, then came the college students, next the major software houses, and now the rest of the programming world. Programmers everywhere are infected with the desire for SPEED, POWER, and PORTABILITY.

It's time to face the inevitable. You're going to catch the fever too! When you do, give us a call. We've got the best cure—an illustrated guide to the C language, plus a complete program development system. Everything you need to master the C programming language... all at a price that's less than the cost of a book!

But don't let this price fool you. Our system is powerful; it compiles twice as fast as the others, is completely standard, and it's very easy to use. Most C compilers are designed for wizards. We have designed ours for you!

What do you get for a mere \$39.95?

- A 450 Page book filled with sample programs, plus...
- A fast, standard, full featured C compiler that supports all data types and the latest features like bit fields, enumerations, structure assignment, and passing/returning structures.
- A fast linker that loads separately compiled files, searches libraries, and builds an executable program.
- An extensive library of over 170 functions (including the standard C functions and the computer specific functions that provide direct access to the operating system and BIOS).
- Works great! Works great! Works great! your programs for minimal space or maximum speed.

**Works on all
MSDOS/PCDOS
and all CP/M Z80
COMPUTERS!
(Not Copy
Protected!)**

**The Powerful MIX
C COMPILER**
Harness the Power of the C Language
with this full featured Compiler

Incredible Value

\$39.95
AT ONLY With 30 Day

Money-Back Guarantee

Operators are standing by... Please use this Number for ORDERS ONLY!

CALL TOLL FREE FOR RUSH ORDER DELIVERY!

1-800-523-9520

**IN TEXAS, PLEASE
CALL TOLL FREE
1-800-622-4070**

For Technical Support Please call 1-214-783-6001

MIX Software, Inc. / 2116 E. Arapaho / Suite 363 / Richardson, Texas 75081

Or contact our Worldwide Distributors direct in:

Canada: Saraguay Software 1-416-923-1500 Switzerland: DMB Communication CH-1-825-53-29
Australia: Techflow 047-586924 France: Info/Tech 1-43-44-06-48

Split Screen Text Editor

**An Incredible Value
AT ONLY \$29.95**

(Not Copy Protected!)
Works on all MSDOS,
PCDOS and CP/M Z80
Computers.

Our high powered editor is great for editing high level languages. It works just like Micropro's Wordstar[®] but macros allow you to create your own custom editor, and the split-screen feature lets you edit two files at the same time.

The MSDOS/PCDOS version is loaded with special features:

- Execute any DOS command or RUN other programs from the editor.
- Quickly edit files as large as 300,000 characters.
- Compile MIX C programs directly from memory. The editor automatically positions the cursor to the first error in your program.

ASM UTILITY

An Incredible Value AT ONLY \$10

Call assembly language routines from your C programs. The ASM utility works with Microsoft's MASM or M80 assembler. Macros make it easy! Works just as if you were calling a C function, and you can even call C functions from assembly language. Lots of useful assembly language functions are included as examples.



SPECIAL OFFER!

Buy both for an even greater value!

Limited Time Only
\$54.95
C Compiler & Text Editor

SAVE \$14.95 Off Our Regular List Price!

Please check method of payment:

☐ Check ☐ Money Order ☐ MasterCard/VISA

Your Card #: _____

Expires _____

Shipping Charges: (No charge for ASM Utility)

In the U.S.A.: Add \$5.00 per Order.

In CANADA: Add \$10.00 per Order.

OVERSEAS: Add \$10.00 per Text Editor. Add \$20.00 per C Compiler. Add \$30.00 for combined C Compiler and Text Editor.

Operating System: (Check one)

☐ CP/M Z80 ☐ MSDOS/PCDOS

Specify Your Computer Name _____

Specify Disk Format _____

NAME _____

Telephone A/C (_____) _____

Street _____

City _____

State _____

Country _____ ZIP _____

MIX software 2116 East Arapaho
Suite 363
Richardson, Texas, 75081

Ask about our Volume Discounts!
Call 1-214-783-6001

Description	Quantity	PRICE	Total Order
Split-Screen Text Editor	_____	\$29.95	\$ _____
C Compiler	_____	\$39.95	\$ _____
C and Text Editor (Special)	_____	\$54.95	\$ _____
ASM Utility	_____	\$10.00	\$ _____
Texas Residents Add 6.125% Sales TAX			\$ _____
Shipping Charges (See at Right)			\$ _____
TOTAL OF YOUR ORDER:			\$ _____

Even at this low price, there is no longer any reason to accept a subset compiler. There are now several full compilers to choose from at similar prices.

Whitesmiths C

Whitesmiths has probably been around longer than any other compiler vendor mentioned here. It provides its compiler for many different machine/operating system combinations. At \$1,000 it is the most expensive compiler we reviewed. The language supported is as close or closer to the ANSI standard (depending on which one you read) than any other compiler's. It supports three separate libraries—ANSI, extended ANSI (the extensions are very useful), and Whitesmiths. Whitesmiths C supports some pretty useful "space modifiers." In addition to the standard *near* and *far*, it also supports a port address space for letting variables point at I/O ports. The major missing features have got to be excellent code quality and a speedy library.

The documentation is complete and includes a language reference manual. The three libraries are described on separate library pages so you won't be looking at a function that is not available in your library. Although the documentation is clearly sectioned so that common sections can be used "as is" for other variations of the compiler, this is not the impediment to usefulness that it could be. The manual does include a style guide with specific suggestions on how to write portable code. Portable code has got to be Whitesmiths best point, supporting C for CP/M, VAXs, PDP-11s, 68000s, and IBM mainframes.

Source-level debugging is provided for in a unique way. You compile your source for *debug*, and when you link it, the source-level debugger is linked in. When you run the program, the debugger is invoked.

Wizard C

Wizard Software Systems was a newcomer to the MS-DOS C compiler market when our review appeared last year, but its compiler was nevertheless one of the top four contenders. Its strongest point was its extensive er-

ror checking and lint feature. This year we received a beta-test copy of Wizard C, Version 3.0, a significantly enhanced compiler. Added are several features from the emerging ANSI standard, support for mixed-model programming, and an improved optimizer that includes an algorithm that automatically allocates register variables if you do not.

Wizard C, Version 3.0, supports programming in nine memory models, with three floating-point-support options. It can generate code for 80186 and 80286 processors. Code generated for the 80286 expects to be run in the protected mode and takes full advantage of 32-bit addressing, so it cannot run under current versions of MS-DOS. In addition to a large range of optional warning messages, the Wizard compiler offers a lint mode of compilation that performs full cross-checking of multiple source files. The full K & R language is implemented, plus *void*, *enum*, *signed*, *const*, *volatile*, structure and union assignment, structure and union pass/return, function prototyping, and several other ANSI features.

The Wizard C compiler comes on four disks and includes the source code for the library. The compiler itself is four passes, but a *cc* driver program is supplied that accepts wild-card file names and runs all passes plus the linker. As with most vendors, Wizard uses the Microsoft object format and linker. The compiler has no assembler as such, although in-line assembly code is allowed—if you want to link to separate assembler routines or do in-line assembly, you will need your own Microsoft-compatible assembler. There are no batch files to automate the installation process, but the manual on package installation and usage includes sections describing floppy- and hard-disk setup.

The compiler is documented in a separate manual with its own table of contents and index. The copy we received was complete and well organized and included a quick-reference chart listing all available options in one place. A lot of options are available—we used eight on the command that tries to generate minimal compile times. There are four optimization options, one of which specifies that the more efficient PL/M calling conven-

tions be used for all functions not taking a variable number of arguments.

You can build a configuration file that contains your most frequently used command-line options. Toggle options turned on in the configuration file can be turned back off for an individual compile by repeating them on the command line. The configuration file may be in the current directory or in any directory on the DOS path. This seems like a more flexible and powerful approach than the use of environment variables.

The documentation for Wizard C consists of three separate manuals, typewritten on 8.5×11-inch paper, in a 1-inch ring binder. The manuals we received, like the compiler, were at the beta-test stage, and we found some rough edges. The 1-inch binder is a little too small for the 400 pages it contains, so the pages tend to bind when you flip around.

The first manual, a reference to compiler installation and usage is complete and well organized, including a table of contents and index. It includes a section on compiler diagnostics, with a paragraph describing each error.

The second manual is a complete language reference and includes a table of contents but no index. This manual is extensive compared to those that most vendors provide, but it is too terse to serve as your only reference to the C language. There is no summary of differences from the K & R definition of C.

The third manual is the library reference and includes a table of contents and an index. The function definitions are organized in Unix fashion, with no examples anywhere. A library summary section provides a one-line description of what each function does, organized by category. This will help you find the function that does what you want more quickly, but it does not serve as a quick reference to the library. Wizard includes a set of screen functions and a Unix emulation package in the library. There are more than the usual number of DOS interface functions and a useful section in the overview that lists the most appropriate library function for each DOS system call. The overview does not contain nearly enough background information on buffered and unbuffered I/O.

PROGRAMMER'S CONNECTION

There is a difference.



Please don't mistake us for our competition. We're Programmer's Connection, a leading independent dealer of quality programmer's development tools specifically for IBM Personal Computers. We're your best one-stop source for the professional PC/MS-DOS and XENIX programming tools you need.

Since we're an *independent* dealer, we'll look out for your best interest. Our courteous, knowledgeable, noncommissioned sales staff is always ready to assist you. If you aren't sure about your needs, you can talk to our experienced, professional technical staff for sound, unbiased advice. They can compare products, answer technical questions and send you detailed product information that's tailored to your needs.

Our product line consists of hundreds of high quality software development tools specifically for IBM Personal Computers and compatibles. We don't carry *everything* ever written for programmers — only those products that meet our very high standards for quality and value.

The products we carry are the latest version and they come with the same manufacturer's technical support as if buying direct. Unlike other dealers who participate in the software gray market, we're authorized to sell *every* product we carry.

We discount all software products — even special order items. We don't try to fool you by discounting some products and charging full list price for others. Every product in our price list is shown with its discounted *and* retail price. We want you to know exactly how much you'll save on *every* product.

Other dealers add extra rush charges for shipping via express services. We'll express your order to you with no special handling charges. We only charge you what the shipping carrier charges. And if you have your order shipped via standard UPS, *we'll* even pay for the shipping!

Most popular products are in stock and ready for immediate shipment. Maintaining an adequate inventory is part of our philosophy of fast, efficient service.

If we don't have a product in stock, we'll get it for you fast. Again — no extra charge. Some dealers charge your credit card at the time they take your order. This means you could be left waiting in vain for your order for weeks or months while they use your money interest free. We *never* charge your credit card until we actually ship your order.

Quite simply, the discount prices you see on the next two pages are all you pay. There are *no* hidden charges. We don't charge extra for standard UPS shipping, credit cards, COD orders, purchase orders or special handling (except orders outside the U.S. and Canada are charged \$5 for customs form preparation).

Our goal is customer satisfaction and that's why we offer 30 day no-risk return guarantees and 30 day evaluation periods on most of our products. Note that some items, especially those with source code, are restricted by the manufacturers from this guarantee. Please call for specific details.

As you can see, we're *not* like other dealers. Our customers keep coming back because we consistently provide high quality service and low discount prices.

So make the connection today and discover the difference for yourself. You'll be glad you did!

Call or write for our *FREE* comprehensive price guide.

US: 800-336-1166

CANADA: 800-225-1166

OHIO AND OVERSEAS: 216-877-3781

Customer Service: 216-877-1110

Hours: 8:30 AM to 8:00 PM Eastern Time.

 **programmer's
connection**

Turn the page for our latest advertised price list. ►



PROGRAMMER DEVELOPMENT TOOLS FOR THE IBM-PC/XT/AT and compatibles.

See previous page
for more information.

apl language

APL*PLUS/PC System by STSC	595	449
APL*PLUS/PC Tools Vol 1 by STSC	295	239
APL*PLUS/PC Tools Vol 2 by STSC	85	69
APL*PLUS Spreadsheet Manager by STSC	195	159
APL*PLUS/UNIX System For AT Xenix by STSC	995	795
Btrieve ISAM File Mgr with No Royalties by SoftCraft	250	195
Financial/Statistical Library by STSC	275	219
Pocket APL by STSC	95	79
QNIAL Combination of APL with LISP by NIAL Systems	375	359
STATGRAPHICS Statistical Graphics System by STSC	795	619

arity products

Arity Expert System Development Package	New	295	279
Arity File Interchange Toolkit	New	50	48
Arity Prolog Compiler & Interpreter	New	795	739
Arity Prolog Interpreter	New	350	329
Arity SQL Development Package	New	295	279
Arity Screen Design Toolkit	New	50	48
Arity Standard Prolog	New	95	89

artificial intelligence

ExpertEDGE by Human Edge	795	659	
Expteach Complete System by Intellware	475	389	
EXSYS Expert System Development Software by EXSYS	395	339	
First Class by Human Edge	New	500	419
GCLISP Golden Common LISP by Gold Hill	495	CALL	
GCLISP 286 Developer LM Interpreter & LM Compiler	1190	CALL	
Insight 1 AI Primer by Level Five Research	95	75	
Insight 2+ by Level Five Research	New	485	389
Logic-Line Series All varieties by Thunderstone	New	CALL	CALL
Microsoft LISP Common LISP	250	189	
PLA microProlog by Programming Logic Associates	250	219	
with APES	450	399	
PLA Professional microProlog by Programming Logic	395	349	
with APES	695	599	
QNIAL Combines APL with LISP by NIAL Systems	375	359	
Turbo Prolog Compiler by Borland International	100	79	

assembly language

8088 Assembler w/Z-80 Translator by 2500 AD	100	89
ASMLIB Function Library by BC Associates	New	149 129
Cross Assemblers Over 25 Varieties from 2500 AD	CALL	CALL
Microsoft Macro Assembler with utilities	150	99
Turbo EDITASM Fast Assembler by Speedware	99	84
Visible Computer: 8088 by Software Masters	80	65

basic language

BetterBASIC by Summit Software	200	165
8087 Math Support	99	85
Btrieve Interface	99	85
C Interface	New	CALL CALL
Run-time Module	250	225
Microsoft QuickBASIC Compiler	New version	99 79
Professional BASIC by Morgan Computing	99	79
8087 Math Support	50	47
True Basic		150 105
Run-time Module	Special Price	500 199

blaise products

Asynch Manager Specify for C or Pascal	175	136
C Tools Combination Package Both Items Below	175	149
C Tools	125	105
C Tools 2	100	84
Exec Program Chainer	95	79
Pascal Tools Combination Package Both Items Below	175	149
Pascal Tools	125	105
Pascal Tools 2	100	84
Turbo ASYNCH for Turbo Pascal	100	84
Turbo POWER TOOLS for Turbo Pascal	100	84
View Manager Specify for C or Pascal	275	209
with Source Code	295	239

borland products

REFLEX Data Base System	150	119	
REFLEX Workshop	New	70	59
REFLEX and REFLEX Workshop Combination Package	New	200	159
Turbo DATABASE TOOLBOX	55	38	
Turbo EDITOR TOOLBOX	70	54	
Turbo GAMEWORKS TOOLBOX	70	54	
Turbo GRAPHIX TOOLBOX	55	38	
Turbo LIGHTNING	99	75	
Turbo PASCAL	70	49	
with 8087 or BCD	110	77	
with 8087 and BCD	125	84	
Turbo Prolog Compiler	100	79	
Turbo TUTOR for Turbo PASCAL	35	28	

c compilers

C-86 Optimizing Compiler	395	289
Dataltight C Compiler <i>Small Memory Model</i>	60	49
Dataltight Developer's Kit <i>with Large Memory Model</i>	99	79
DeSmet C Compiler <i>with Source Debugger</i>	159	145
Large Case Option	New	50
Eco-C <i>Complete Development System by Ecosoft</i>	125	89
Lattice C Compiler <i>from Lattice</i>	500	299
See Lattice Section		
Let's C Compiler <i>by Mark Williams</i>	75	59
with csd Source Level Debugger	150	118
Microsoft C Compiler <i>with Source Debugger</i>	New version	CALL
MWC-86 <i>by Mark Williams</i>	495	299
Wizard C Compiler <i>Includes Lint by Wizard Systems</i>	450	369

c interpreters

C-terp by Gimpel Software Specify compiler interface	300	239
Instant C by Rational Systems	500	379
Introducing C by Computer Innovations	125	105
Run/C by Age of Reason	150	99
Run/C Professional by Age of Reason	250	184

c utilities

Also refer to Blaise, Computer Innovations, Lattice, Microsoft, Phoenix, Polytron, SoftCraft and Xenix System V sections.

APT Application Programmer's Toolkit by Shaw American	395	339
Basic C Library by C Source	175	135
C Essentials by Essential Software	100	85
C to dBase ISAM Manager by Computer Innovations	150	139
c-tree ISAM File Manager with source by FairCom	395	329
C Utility Library by Essential Software	185	139
C Windows by Syscom	100	89
C Wings by Syscom	50	45
CI Probe Source Level Debugger Limited Quantity Sale	225	159
CI RomPac for C-86 by Computer Innovations	195	149
dBx dBase to C Translator by Desktop AI New	350	329
dbVISTA Single-User DBMS by Raima	195	159
with Source Code	495	429
dbVISTA Multi-User DBMS by Raima	495	429
with Source Code	990	849
Entelekon Combo Package All 3 items below	200	175
C Function Library	130	115
C Windows	130	115
Superfonts for C	50	45
Essential Graphics No Royalties by Essential Software	250	219
Flash-up Windows by Software Bottling of NY	75	69
Graphic Mono version 2.2 by Scientific Endeavors	280	219
Graphic Color version 3.0 by Scientific Endeavors	350	299
The Greenleaf Functions by Greenleaf Software	185	135
Greenleaf Comm Library by Greenleaf Software	185	135
The HAMMER by OES Systems	195	175
MetaWINDOWS by Metagraphics	185	139
MetaWINDOWS/Plus by Metagraphics	235	199
Multi-Halo with Royalties by Media Cybernetics	300	219
On-line Help from Opt-Tech Data Processing	149	119
PANEL by Roundhill Library Source Also Available	295	229
PC Lint by Gimpel Software	139	109
Scientific Subroutine Library for C by Peerless	175	139
Vitamin C by Creative Programming	150	139
VC Screen Interactive Forms Designer	99	85
Zview with Free Updates by Data Mgmt Consultants	245	199

cobol language

Micro Focus COBOL Workbench	4000	3599
Micro Focus Level II COBOL	Special Price 1500	CALL
COMATH	200	169
FORMS-2	300	269
Level II Animator	Special Price 900	CALL
Level II SOURCEWRITER	Special Price 2000	CALL
Micro Focus Level II COBOL For Novell NetWare	New 2000	1799
Micro Focus Micro/SPF	175	159
Micro Focus Professional COBOL	3000	2395
Multi-user Runtime for PC Network	New 500	449
Microsoft COBOL	See Microsoft Section	
OPT-Tech Sort Also Sorts Btrieve Files	149	119
Realia COBOL	995	795
RM/COBOL by Ryan-McFarland	950	675
RM/COBOL 8X ANSI R5 COBOL by Ryan-McFarland	1250	995

debuggers & profilers

Advanced Trace-86 with ASM Interpreter by Morgan	175	139
CI Probe Source Level Debugger	Limited Quantity Sale	225 159
Codesifter Execution Profiler by David Smith	New	119 99
Codesmith-86 Debugger by Visual Age		145 109
Periscope I w/Board & Switch by Data Base Decisions		295 249
Periscope II w/NMI Breakout Switch Only		145 115
Periscope II-X Software only	Special Price thru August	115 85
The PROFILER with Source Code by DWB associates		125 95

forth language

CFORTH Native Code Application Compiler by LMI	New	300	239
LMI Forth/83 Metacompiler Specify Target Processor	New	750	599
PC/Forth by Laboratory Microsystems		150	119
PC/Forth+ by Laboratory Microsystems		250	209
Advanced Color Graphics Support	New	100	79
Enhanced Graphics Support	New	200	159
Intel 8087 Support	New	100	79
Interactive Symbolic Debugger	New	100	79
Native Code Optimizer	New	200	159
PCTERM Modem Program for Smartmodem	New	100	79
Software Floating Point	New	100	79

fortran language

ACS Time Series by Alpha Computer Service	495	429
Btrieve ISAM File Manager	See SoftCraft Section	
50 MORE: FORTRAN by Peerless Engineering	125	99
For-Winds by Alpha Computer Service	90	79
Forlib-Plus by Alpha Computer Service	70	55
Grafmatic or Plotmatic by Microcompatibles	135	119
Grafmatic and Plotmatic by Microcompatibles	240	219
Microsoft Fortran	350	209
Multi-Halo with Royalties by Media Cybernetics	300	219
PANEL Screen Designer by Roundhill	295	229
RM/Fortran by Ryan-McFarland	595	395
Scientific Subroutine Library by Peerless	175	139
The Statistician by Alpha Computer Service	295	259
Strings & Things by Alpha Computer Service	70	55

lattice products

Lattice C Compiler from Lattice	500	299
with Library Source Code	900	549
C Cross Reference Generator	50	39
with Source Code	200	159
C-Food Smorgasbord Function Library	150	99
with Source Code	300	195
C-Sprite Source Level Debugger	175	139
Curses Screen Manager	125	98
with Source Code	250	194
dBC dBase File Manager for C	250	194
with Source Code	500	388
LMK Make Facility	195	149
RPG II Compiler No Royalties	750	595
SecretDisk File Encryption Utility	60	49
SideTalk Resident Communications	120	95
Text Mgmt Utilities (GREP,DIFF,ED,WC,Extract,Build)	120	95
TopView Toolbasket Function Library	250	194
with Source Code	500	388
Z-80 C Cross Compiler	500	388
with Library Source Code	1000	789

microsoft products

Microsoft BASIC Interpreter for Xenix	350	279
Microsoft C Compiler with source debugger	CALL	CALL
Microsoft COBOL Compiler	700	495
for Xenix	995	795
Microsoft COBOL Tools with COBOL Source Debugger	350	209
for Xenix	450	359
Microsoft Fortran Compiler	350	209
for Xenix	495	389
Microsoft LISP Common LISP	250	189
Microsoft Macro Assembler with utilities	150	99
Microsoft Mouse Bus Version	175	149
Microsoft Mouse Serial Version	195	159
Microsoft muMath Includes muSIMP	300	195
Microsoft Pascal Compiler	300	195
for Xenix	495	389
Microsoft QuickBASIC Compiler	99	79
Microsoft Sort	195	149
Microsoft Windows	99	74
Microsoft Windows Developer's Kit	500	CALL

modula-2 language

MODULA-2/86 Compiler by Logitech	89	65
with 8087	129	105
with 512K	189	149
MODULA-2 Editor by Logitech	59	49
MODULA-2 Runtime Debugger by Logitech	69	59
MODULA-2 Source Package by Logitech	99	79
MODULA-2 Utilities Package by Logitech	49	45

other languages

CCS MUMPS Single-User version by MGlobal	60	55
CCS MUMPS Multi-User version by MGlobal	450	379
Janus/ADA C Pack by R&R Software	95	89
Janus/ADA D Pack by R&R Software	900	699
Methods Smalltalk by Digitalk	79	69
Smalltalk IV by Digitalk	99	89
SNOBOL4+ by Catspaw	95	84

other products

Dan Bricklin's Demo Program by Software Garden	75	65
FASTBACK Backup Utility by 5th Generation Systems	179	159
Interactive EASYFLOW by Haventree Software	150	129
SET:SCIL by System Engineering Tools	349	299
Source Print by Aldebaran Laboratories	139	119
SRMS Software Revision Mgmt System by Quilt Computing	125	109

phoenix products

Plantasy Pac (Pfinsh,Plix+,Plinkr,Pmaker,Pmate,Ptel)	1295	949
Pfinsh Performance Analyzer	395	249
Plix-86 Plus Symbolic Debugger	395	249
PforCe Comprehensive C Function Library	395	289
Plink-86 Overlay Linker	395	249
Plink-86 Plus Enhanced Overlay Linker	495	389
Pmate Macro Text Editor	225	149
Pre-C Lint Utility	395	249
Ptel Binary File Transfer Program	195	139

polytron products

Polytron C Beautifier	49	45
Polytron C Library I	99	79
Polytron PowerCom Communications	179	139
PolyLibrarian Library Manager	99	79
PolyLibrarian II Library Manager	149	119
PolyMake UNIX-like Make Facility	99	79
PolyOverlay Overlay Optimizer	99	79
PolyWindows Products All Varieties	CALL	CALL
PolyXREF Complete Cross Reference Utility	219	179
PolyXREF Support for one language only	129	109
PVCS Polytron Version Control System	395	329
PVMFM Polytron Virtual Memory File Manager	199	149

softcraft products

Btrieve ISAM File Manager with No Royalties	250	195
Xtrieve Query Utility for Btrieve	195	169
Rtrieve Report Generator for Xtrieve	85	79
Btrieve/N for Networks	595	465
Xtrieve/N Query Utility for Btrieve/N	395	299
Rtrieve/N Report Generator for Xtrieve/N	175	159

text editors

Brief from Solution Systems	195	CALL
Epsilon Multi-tasking Emacs-like editor by Lugaru	195	165
FirstTime for Turbo by Spruce Technology	75	69
KEDIT Xedit-like editor by Mansfield Software Group	125	109
PC/VI by Custom Software Systems	149	129
SPF/PC by Command Technology Corp	195	165
Vedit by CompuView	150	115
Vedit Plus by CompuView	225	180
XTC Text Editor with source by Wendin	99	84

turbo pascal utilities

Also refer to Blaise, Borland and SoftCraft sections.

ALICE Turbo Pascal Interpreter by Software Channels	95	69
FirstTime for Turbo by Spruce Technology	75	69
Flash-up Windows by Software Bottling of NY	75	69
Multi-Halo with Royalties by Media Cybernetics	CALL	CALL
On-line Help from Opt-Tech Data Processing	149	119
Screen Sculptor by Software Bottling of NY	125	95
Turbo EXTENDER by TurboPower Software	85	69
Turbo Professional by Sunny Hill Software	70	49
TurboPower Utilities by TurboPower Software	95	84
TurboWINDOW by MetaGraphics	80	69

wendin products

Operating System Toolbox Build your own OS	99	84
PCUNIX Operating System	99	84
PCVMS Operating System Similar to VAX/VMS	99	84
XTC Text Editor with Pascal Source Code	99	84

xenix system v

Complete Xenix System by SCO All 3 items below	1295	1099
Xenix Development System	595	529
Xenix Operating System Specify XT or AT	595	529
Xenix Text Processing Package	195	179

xenix languages and utilities

APL*PLUS/UNIX System For AT Xenix by STSC	995	795
Btrieve ISAM File Manager by SoftCraft	595	465
c-tree ISAM File Manager with Source by FairCom	395	329
dBx dBase to C Translator w/source by Desktop AI	550	499
dbvISTA Single and Multi User versions by Raima	CALL	CALL
Informix by Relational Database Systems	995	839
Informix4GL by Relational Database Systems	CALL	CALL
InformixSQL by Relational Database Systems	995	839
Lyrix by SCO	595	489
Micro Focus Level II Compact COBOL For AT	1000	899
Forms-2	400	359
Level II ANIMATOR	600	539
Microsoft Languages	See Microsoft Section	
RM/COBOL by Ryan-McFarland	1250	995
RM/FORTRAN by Ryan-McFarland	750	599
SCO Professional Complete Lotus clone by SCO	795	695



Prices are subject to change without notice.

Ohio customers please add 5% state sales tax.



Hours: 8:30 AM to 8:00 PM EST, Monday through Friday.



U.S.

1-800-336-1166



CANADA

1-800-225-1166



OHIO AND OVERSEAS

1-216-877-3781

CUSTOMER SERVICE 216-877-1110

programmer's connection

 136 SUNNYSIDE ST.
HARTVILLE, OHIO 44632

Hot C

WordTech Systems markets a compiler for dBASE III. Recently (by the time you read this), it has started to sell a C compiler imported from Hi-Tech Software of Australia. WordTech is calling the package Hot C and is apparently marketing it primarily as a complement to its dBASE compiler. We received a preliminary version of the package for review.

Hot C allows programming in the large/large-memory model and can produce programs that make use of an 8087. It does not include a sensing library, so a program that uses an 8087 will not run if an 8087 is not present. Hot C does not implement bit fields, but *void*, *enum*, structure assignment, and structure pass/return are supported. It includes a symbolic (not source-level) debugger, somewhat similar to the Unix debugger *adb*. WordTech claims that an editor and a C language tutorial will be included with the final package, but they were not available when we received the package. The version of the compiler we received included a very preliminary copy of the documentation, so we cannot comment on the quality of the package you will receive. The compiler itself comes on two disks, with the source for the library in compressed format.

Hot C does not support a separate environment variable to specify the search path for include files, as do most compilers. It does support an environment variable to specify where to store temporary files and where to find the compiler passes. The latter option is not optional if you have a hard disk. The driver program looks for the compiler on drive A: if you do not tell it otherwise—it does not scan the path or the current directory! Likewise, drive A: is searched if the preprocessor cannot find one of your include files in the current directory or one of the directories you specify with *-I* arguments on the command line.

The preliminary copy of the Hot C manual we received had several problems, including page layout, that we assume will be rectified before WordTech actually starts shipping. We are more concerned about the

contents than the format. Our copy had no index or table of contents, and most of the manual consists of a series of appendices. There are also frequent references to a Z80 version of the compiler that may not even be available in this country.

The first appendix is a detailed list of differences from the K & R C "standard." It is organized to correspond to the Reference Manual section of K & R on a point-for-point basis and is complete. This is the right way to present this information.

The library reference is patterned after the Unix model, with brief text describing several functions under a single heading and no examples. This kind of format is difficult to use, especially in the absence of an index, because functions are not in alphabetical order. Related functions are cross-referenced, however, and there is a quick-reference list of functions organized by category. Sometimes information is missing—for instance, the DOS interrupt function *msdos-cx()* returns a *long* containing registers *cx* and *dx* returned from DOS, but whether *dx* or *cx* is in the high-order word of the *long* result is not mentioned (*dx* is).

Summary

We promised to draw some conclusions about these results, based on both what we have written and what we could not write because of time and space limitations.

Datalight's products are the best value in MS-DOS C compilers today. At \$60 for a full K & R compiler (Datalight C), you get a fast-compiling, easy-to-use, good code-quality compiler. And it is compatible with Lattice's calling convention to boot. For \$99 total (for the Datalight C Developer's Kit), you can add three more memory models and the source code for the entire run-time system.

Microsoft's compiler is the best MS-DOS C development environment value today. At \$395 list (less than \$300 discounted), you get everything you could want (except an editor and assembler) to develop virtually any kind of program conceivable. A make facility, the compiler, a linker, and a debugger allow generation of high-quality, unlimited-size programs with almost total flexibility.

Wizard's is the best compiler to

day. What it does have is library source for a very large library, good documentation, excellent support, and lint.

Our choice if we could make our own? We would take Wizard's compiler (with the huge keyword and MetaWare's pragmas added) and Microsoft's library (with source), documentation, and debugger at Datalight's price.

We are also strongly attracted to MetaWare's High C, Manx Aztec C and DeSmet C by C Ware. High C is the most flexible compiler available (if you need it, High C has got it), has very complete documentation, and has absolutely the best support. Code quality is very good, the compiler and library are fully compatible with the emerging ANSI standard, and compiler diagnostics are the best. You can move code between this compiler and corresponding compilers for the IBM RT/PC, the Atari, IBM mainframes, DEC VAX, and 68000 and 32000 series processors. However, it is a slow compiler, needs lots of room, costs \$495, and does not include full library source or a debugger.

Aztec C does include full library source and a source-level debugger. The code quality is good with promises to get better in the next release. It includes an editor, make, assembler, and linker, and the linker can locate, allowing easy ROM development. Moving code across different micros is easy because Manx has compilers for Apple, CP/M, MS-DOS, the Amiga, the Macintosh, and Atari computers. But the documentation is difficult to use, the price is \$495, and the code quality, although good, is not among the best anymore.

DeSmet C is another value-packed system, including an editor, assembler, and linker. Library source, a good source-level debugger, speedy compiles, and a low price distinguish this compiler from the others. Minor improvements in the library, debugger, code generator, and manual would make this package hard to beat.

DDJ

(Listings begin on page 104.)

Vote for your favorite feature/article.
Circle Reader Service No. 3.

C Programmers! First database written exclusively for C is also royalty free

"If you are looking for a sophisticated C Programmer's Database, db_VISTA™ is it..."

Dave Schmitt, President, Lattice, Inc.

Designed exclusively for C, db_VISTA™ is a royalty-free programmer's DBMS. Both single and multi-user versions let you take full advantage of C, through ease of use, portability and efficiency.

Written in C for C Programmers

All functions use C conventions so you will find db_VISTA easy to learn. db_VISTA operates on most popular computers, and because it is written in C it can easily be ported to most computers.

Royalty-Free, You only pay once

Whether you're developing applications for a few customers, or for thousands, the price of db_VISTA is the same. If you are currently paying royalties for a competitor's database, consider switching to db_VISTA and say goodbye to royalties. To help you make the change over to db_VISTA, ASCII file transfer utilities are included. dBASE file transfer utilities are available as an option.

More from your database applications with source code

Source code includes all db_VISTA libraries and utilities.

1. Recompile our run-time libraries utilizing non-standard compiler options.
2. Create a debugging library including a function traceback by activating pre-processor commands embedded in the source code.

Multi-user and LAN capability

Information often needs to be shared. db_VISTA has multi-user capability and supports simultaneous users in either multi-tasking or local area networking environments, allowing the same C applications to run under UNIX and MS-DOS.

Faster execution without data redundancy

Less data redundancy means reducing disk storage requirements and maximizing data access performance. A customer evaluating a leading competitor's product prior to purchasing db_VISTA benchmarked db_VISTA's retrieval time to be 276% faster than a leading competitor.

Complete documentation included

User manual contains 193 pages, 8 diagrams, 10 tables, appendices, an extensive index, plus a database application example. 9 chapters with complete instructions.

Introducing db_QUERY™

With db_QUERY you can ask more of your database, db_QUERY is a linkable, SQL-based ad hoc query and report writing facility. It's also royalty-free.

30 day Money-Back Guarantee

We wish to give you the opportunity to

try db_VISTA for 30 days in your development environment and if not satisfied return it for a full refund.

SAVE \$50 TO \$100! PURCHASE db_VISTA™ BY AUGUST 31, 1986 and, RECEIVE UP TO \$100 REBATE*

Price Schedule

	db_VISTA	Rebate
Single-user	\$195	
Single-user with Source	\$495	\$ 50
Multi-user	\$495	\$ 50
Multi-user with Source	\$990	\$100

Free 90 days application development support
All software Not Copy Protected

Call Toll Free Today and Learn How To Receive Up To \$100

To order or for information, call TOLL FREE 1-800-843-3313, at the tone touch 700-992 or call 206-747-5570.
VISA and MASTERCARD Accepted

Read what others say about db_VISTA...

"If you are looking for a sophisticated C programmers database, db_VISTA is it. In either a single or multi-user environment, db_VISTA lets you easily build complex databases with many interconnected record types. The multi-user implementation handles data efficiently with a LAN and Raima's customer support and documentation is excellent. Source code availability and a royalty-free run-time is a big plus."

Dave Schmitt, President
Lattice, Inc.

"Not yet another user-friendly database", it is a DBMS aimed at the technical C programmer instead of the non-technical end-user."

Hal Schoolcraft, Data Based Advisor
March, 1985

"On the whole, I have found db_VISTA easy to use, very fast with a key find, and powerful enough for any DBMS use I can imagine on a microcomputer."

Michael Wilson, Computer Language
September, 1985

db_VISTA Version 2.11 Database Management System for C

Database Record and File Sizes

- Maximum record length limited only by accessible RAM
- Maximum records per file is 16,777,215
- No limit on number of records or set types
- Maximum file size limited only by available disk storage
- Maximum of 255 index and data files

Keys and Sets

- Key length maximum 246 bytes
- No limit on maximum number of key fields per record - any or all fields may be keys with the option of making each key unique or duplicate
- No limit on maximum number of fields per record, sets per database, or sort fields per set
- No limit on maximum number of member record types per set

Utilities

- Database definition language processor
- Interactive database access utility
- Database consistency check utility
- Database initialization utility
- Multi-user file locks clear utility
- Key file build utility
- Data field alignment check utility
- Database dictionary print utility
- Key file dump utility
- ASCII file import and export utility

Features

- Multi-user support allows flexibility to run on a local area network
- File structure is based on the B-tree indexing method and the network database model
- Run-time size, variable - will run in as little as 64K, recommended RAM size is 256K
- Transaction processing assures multi-user database consistency
- File locking support provides read and write locks on shared databases
- SQL-based db_QUERY is linkable
- File transfer utilities included for ASCII, dBASE optional

Operating System & Compiler Support

- Operating system's MS-DOS, PC-DOS, Unix, Xenix, Macintosh & Amiga
- C compiler's Lattice, Microsoft, DeSmet, Aztec, Computer Innovations, Xenix and Unix

Independent Benchmark Results

Eleven key retrieval tests on sequentially and randomly created key files. Benchmark procedure adapted from "Benchmarking Database Systems: A Systematic Approach" by Bitton, DeWitt, and Turbyfill, December, 1983

Total Retrieval Time of 11 Tests	
db_VISTA	:671.24
Leading competitor	:1,856.43



12201 S.E. Tenth Street
Bellevue, WA 98005 USA
(206) 747-5570
Telex: 9103330300 BCN RIVERTON

1 (800) 843-3313

at the tone touch 700-992



* Limited offer available to end-user purchases directly from Raima Corporation.

Circle no. 206 on reader service card.

The fastest C

Your search for execution speed is over. The new Microsoft® C Compiler Version 4.0 is here. With blazing performance. We've added common sub-expression elimination to our optimizer that produces code that rips through the benchmarks faster than ever before.

"...the Microsoft performance in the benchmarks for program execution is the best of the lot overall."
— William Hunt, *PC Tech Journal*, January, 1986*

But speed isn't the only edge you get with Microsoft C. Other advantages include a variety of memory models like our new HUGE model that breaks the 64K limit on single data items. Plus our NEAR, FAR and HUGE pointers, which provide you greater flexibility. All this allows you to fine tune your program to be as small and fast as possible.

"Excellent execution times, the fastest register sieve, and the best documentation in this review ... Microsoft Corporation has produced a tremendously useful compiler." — Christopher Skelly, *Computer Languages*, February, 1986.

No more debugging hassles. Introducing CodeView. Free.

Now, for a limited time, we'll give you an unprecedented programming tool when you buy Microsoft C, free. New Microsoft CodeView™ offers the most powerful tool yet in

the war on C bugs. Forget the hex dumps. Now you can view and work with programs at any level you want. Use the program source, the disassembled object code, or

Microsoft C Compiler Version 4.00

Microsoft C Compiler

- Produces fast executables and optimized code including elimination of common sub-expressions. NEW!
- Implements register variables.
- Small, Medium and Large Memory model libraries.
- Compact and HUGE memory model libraries. NEW!
- Can mix models with NEAR, FAR and the new HUGE pointers.
- Transport source and object code between MS-DOS® and XENIX® operating systems.
- Library routines implement most of UNIX™ System V C library.
- Start-up source code to help create ROMable code. NEW!
- Full proposed ANSI C library support (except clock). NEW!
- Large number of third party support libraries available.
- Choose from three math libraries and generate in-line 8087/80287 instructions or floating point calls:
 - floating point emulator (utilizes 8087/80287 if installed).
 - 8087/80287 coprocessor support.
 - alternate math package — extra speed without an 8087/80287.
- Link your C routines with Microsoft FORTRAN (version 3.3 or higher), Microsoft Pascal (version 3.3 or higher) or Microsoft Macro Assembler.
- Microsoft Windows support and MS-DOS 3.1 networking support.
- Supports MS-DOS pathnames and input/output redirection.

Microsoft Program Maintenance Utility. NEW!

- Rebuilds your applications after your source files have changed.
- Supports macro definitions and inference rules.

Other Utilities

- Library Manager.
- Object Code Linker.
- EXE File Compression Utility.
- EXE File Header Utility.

C Benchmarks

	In seconds				
	Microsoft C 4.0	Lattice C 3.0	Computer Innovation C 2.3	Aztec C86 3.2	Wizard C 3.0
Sieve of Eratosthenes (register)	82.9	151.4	172.3	88.0	91.9
Copy Block	86.9	231.7	199.0	123.8	189.5

Run on an IBM PC XT with 512K memory

Microsoft CodeView

Window-oriented source-level debugger. NEW!

- Watch the values of your local and global variables and expressions as you debug.
- Set conditional breakpoints on variables, expressions or memory; trace and single step.
- Watch CPU registers and flags as you execute.
- Effectively uses up to four windows.
- Debug using your original source code, the resulting disassembly or both intermingled.
- Use drop-down menus to execute CodeView commands.
- Access the on-line help to lead you through CodeView's options and settings.
- Easily debug graphics-oriented programs since program output is kept separate from debugger output.
- Keyboard or optional mouse support.
- Enter in familiar SYMDEB or DEBUG commands.



*Reprinted from *PC Tech Journal*, January 1986, copyright 1986, Ziff-Davis Publishing.

you've ever seen.

both at the same time. Open a window to view CPU registers and flags. Watch local and global variables as well. All while your program is running.

CodeView gives you complete control. Trace execution a line at a time—using source or assembly code. Or set conditional breakpoints on variables, memory or expressions. CodeView supports the familiar SYMDEB command syntax, as you'd expect. Commands are also available through drop-down menus. Combine the new window-oriented interface with our on-line help and debugging has never been easier. Or quicker.

Take the \$5 CodeView tour.

You may find it hard to believe our debugger can do all we've claimed. So we're offering test drives. Five bucks will put you behind the wheel of a Microsoft C demo disk with CodeView. See for yourself how fast debugging can get.

For more information about the CodeView demo disk, the new Microsoft C Compiler, a list of third party library support or the name of your nearest Microsoft dealer, call (800) 426-9400. In Washington State and Alaska, (206) 882-8088. In Canada call (416) 673-7638.

Microsoft® C Compiler

The High Performance Software

Microsoft, MS-DOS and XENIX are registered trademarks and CodeView is a trademark of Microsoft Corporation. UNIX is a trademark of AT&T Bell Laboratories. IBM is a registered trademark of International Business Machines Corporation.

AT LAST: Professional Typesetting Capability For PC Users

With **PCTEX**TM — the best-selling full implementation of Professor Don Knuth's revolutionary typesetting program **TEX**.

FINEST Typeset Quality Printing From:

dot matrix laser phototypesetter

$$\sum_{i=1}^{\infty} \frac{1}{i} \begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \int_{-\infty}^{\infty} e^{-x^2} dx$$

WIDEST Range Of Output Device Drivers:

- Epson FX, LQ
- HP LaserJet*
- Toshiba
- Apple LaserWriter
- Corona LP-300*
- APS-5 phototypesetter
- Screen preview, with EGA or Hercules card

MOST COMPLETE Product Offering:

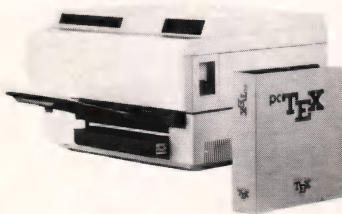
PC_{TEX} (not copy protected) includes the following:

- Our specially written *PC_{TEX} Manual*, which enables you to start using **TEX** right away.
- Custom "macro packages" that provide formats for letters, manuals, technical documents, etc.
- The **L_ATEX** document preparation system, a full-featured macro package for preparing articles, books, reports, etc., and **L_ATEX** User's Manual.
- **AMS-TEX**, developed by the *Amer. Math. Society* for professional mathematical typesetting.

Site licenses, volume discounts, and interfaces to PC Paintbrush, PC Palette, FancyFont and Fontrix are also available.

PRICED FROM ONLY \$249.00!

(Printer drivers and interfaces additional.)



**Laser printer,
fonts & software
from \$2995.00**

For IBM PC/XT, AT or compatible, DOS 2.0 or higher, and 512K RAM. Hard disk required for printer drivers and fonts.
*HP LaserJet and Corona require additional interface boards.

**For more information call or write:
Personal TEX, Inc.**

20 Sunnyside, Suite H, Mill Valley, CA 94941 (415) 388-8853

This ad, with space for the photograph, produced by PC_{TEX}. Typeset on the Epson FX80, the Corona LP-300 laser printer, and the Autologic APS-5 phototypesetter.

TEX is a trademark of the American Mathematical Society. Manufacturers' product names are trademarks of individual manufacturers.

16-BIT

Listing One (text in July)

Screen # 0
(Support for Intel/Lotus Expanded Memory 13:16 08/14/85)

This file contains some simple definitions to allocate Expanded Memory space and use it for word arrays.

The usage within a PC/FORTH program would follow the sequence
EM-OPEN (in program initialization code)
... d @EM ... (various array accesses)
EM-CLOSE (de-allocate memory)

If you fail to issue the EM-CLOSE, the Expanded Memory pages will not be de-allocated and other programs may not be able to obtain sufficient memory.

Copyright (c) 1985 Ray Duncan, Laboratory Microsystems Inc.
P. O. Box 10430, Marina del Rey, CA 90295

Screen # 1
(arrays & variables 13:16 08/14/85)

FORTH DEFINITIONS HEX

(guaranteed device name for)
CREATE em_name , " EMMXXXX0" 0 C, (Expanded Memory Manager)

67 CONSTANT em_int (hex interrupt number for EMM)

-->

Screen # 2
(test for EMM device driver header method 14:02 08/14/85)

```
( --- status ; =0 if EMM present, -1 if not present )
( compares name in presumed device driver to guaranteed name )
CODE ems?
  SI PUSH
  DI, DI XOR ES, DI MOV ( pick up EMM )
  DI, # em_int 4 * MOV ( int vector )
  ES: ES, 2 [DI] MOV DI, # 0A MOV
  SI, # em_name MOV CX, # 8 MOV
  CLD REPZ BYTE CMPS ( compare driver name )
  SI POP
  1$ JNZ ( jump if EMM driver found )
  AX, # 0 MOV 2$ JMP ( return FALSE flag )
  1$: AX, # -1 MOV ( return TRUE flag )
  2$: AX PUSH NEXT, END-CODE
```

-->

Screen # 3
(get EMM frame, EMM free pages 14:02 08/14/85)

```
( --- segment ; get segment of the EMM page frame )
( segment is returned as 0 if function failed )
CODE em_frame AH, # 41 MOV
  em_int INT
  AH, AH OR 1$ JZ BX, # 0 MOV
  BX PUSH NEXT, END-CODE
  1$:
```

```
( returns number of EMM pages which are currently available )
( --- free_pages total pages )
CODE em_pages AH, # 42 MOV em_int INT
  AH, AH OR 1$ JZ DX, # 0 MOV BX, DX MOV
  1$: BX PUSH DX PUSH NEXT, END-CODE
```

-->

Screen # 4
(open EMM & allocate pages 15:45 08/14/85)

```
( get an EMM handle and allocate EMM logical pages to it )
( pages --- handle | 0 )
CODE em_open BX POP
  AH, # 43 MOV em_int INT
  AH, AH OR 1$ JZ ( jump if no error )
  DX, # 0 MOV ( if error return 0 )
  1$: DX PUSH
  2$: NEXT, END-CODE
```

-->

Screen # 5
(memory mapping 16:20 08/14/85)

(continued on page 72)

PLUM HALL

CENTER FOR C LANGUAGE

1 Spruce Av
Cardiff NJ 08232
609-927-3770

- **TRAINING**

Inhouse and Public offerings	
C Programming Workshop	5 days
Preparing for ANSIC	2 days
Advanced C Topics Seminar	3 days

- **PUBLICATIONS**

Learning to Program in C	\$25.00
Reliable Data Structures in C	\$25.00
Efficient C	\$25.00
C Programming Guidelines	\$25.00

- **CONSULTING**

Projects and Coding

- **THE PLUM HALL VALIDATION SUITE FOR C**

Tests accuracy of compilers	
Conformance to the ANSI C Standard	\$9500.

Call or write for further information:

PLUM HALL
1 Spruce Av
Cardiff NJ 08232
609-927-3770
telex 70-3161 PLUMHALL UD

Listing One (Listing continued)

```
( map an EMM logical page owned by handle to a physical page )
( logical_page physical_page handle --- status; =0 if ok )
CODE em_map      DX POP      ( handle )
                  AX POP      ( physical page # )
                  BX POP      ( logical page # )
                  AH, # 44 MOV  em_int INT
                  AH, AH OR 1$ JZ  ( jump if no error )
                  AH, AL XCHG    ( AL := error code )
                  AH, # 0 MOV  2$ JMP
1$: AX, # 0 MOV      ( return 0 if no error )
2$: AX PUSH NEXT, END-CODE
```

-->

```
Screen # 6
( release page allocation                      16:35 08/14/85 )

( release an EMM handle and all logical pages allocated to it )
( handle --- status )
CODE em_close  DX POP      AH, # 45 MOV
                em_int INT
                AH, AH OR 1$ JZ  ( jump if no error )
                AH, AL XCHG    ( AL := error code )
                AH, # 0 MOV  2$ JMP
1$: AX, # 0 MOV      ( return 0 if no error )
2$: AX PUSH NEXT, END-CODE
```

-->

```
Screen # 7
( EM variables & misc defs                      11:13 10/03/85 )

2VARIABLE $EM_USED      ( bytes of EM assigned to arrays )
```

Does this look familiar?
What if each change
you made to your
program was ready to
test in seconds instead
of minutes?



"The SLR tools will change the way you write code. I don't use anything else.", Joe Wright

RELOCATING MACRO ASSEMBLERS • Z80 • 8085 • HD64180

- Generates COM, Intel HEX, Microsoft REL, or SLR REL
- Intel macro facility
- All M80 pseudo ops
- Multiple assemblies via command line or indirect command file
- Alternate user number search
- ZCPR3 and CP/M Plus error flag support. CP/M 2.2 submit abort
- Over 30 user configurable options
- Descriptive error messages
- XREF and Symbol tables
- 16 significant characters on labels (even externals)
- Time and Date in listing
- Nested conditionals and INCLUDE files
- Supports math on externals

requires Z80 CP/M compatible systems with at least 32K TPA

\$49.95

SLR Systems

1622 N. Main St., Butler, PA 16001
(412) 282-0864 (800) 833-3061

Circle no. 78 on reader service card.

```
VARIABLE $EM_PID      ( handle for FORTH from EM manager )
VARIABLE $EM_FRAME    ( segment of EM paging frame )
```

```
( d1 --- d2 ; quick double number multiplies )
: D2* 2DUP D+ ;
: D4* D2* D2* ;
: D8* D4* D2* ;
```

```
( d --- ; compile a double number )
: D, HERE 2! 4 ALLOT ;
-->
```

```
Screen # 8
( EM array alignment, EM-CLOSE                      14:02 10/04/85 )
```

```
( --- ; align EM on 2-byte boundary for single int array )
: WALIGN      $EM_USED 2@ OVER 1 AND 0 D+
              $EM_USED 2! ;
```

```
( --- ; release Expanded Memory allocation )
: EM-CLOSE    $EM_PID @ DUP 0= ABORT" EM not opened"
              em_close ABORT" Can't release memory"
              $EM_PID OFF ;
```

-->

```
Screen # 9
( EM-OPEN                      10:01 10/04/85 )
```

```
( --- ; establish availability of Expanded Memory & allocate )
: EM-OPEN     $EM_PID @ IF EM-CLOSE THEN
              ems? ABORT" Memory manager not installed"
              em_frame DUP 0= ABORT" EM Frame not valid"
              $EM_FRAME !      ( save EM paging segment )
              em_pages SWAP DROP 4000 UM*
              $EM_USED 2@ D<
              ABORT" Insufficient expanded memory available"
              $EM_USED 2@ 4000 UM/MOD SWAP
              IF I+ THEN      ( round up to next page )
              em_open DUP 0= ABORT" Can't get EMM handle"
              $EM_PID ! ;
```

-->

```
Screen # 10
( EM-ARRAY      word array                      12:49 10/03/85 )
```

```
( d_cells --- ;      compiling )
( d_cell# --- em_daddr ; executing )
: EM-ARRAY CREATE WALIGN      ( word align EM ptr )
                    2DUP 1. D- D,      ( highest cell # )
                    D2*      ( *2 for # of bytes needed )
                    $EM_USED 2@      ( get current EM offset )
                    2DUP D,      ( save it )
                    D+ $EM_USED 2!      ( update EM offset )
DOES> DUP >R 2@      ( get # of cells declared )
        2OVER DU< ABORT" Index out of bounds"
        D2*      ( cell# *2 for offset )
        R> 4 + 2@ D+ ;      ( + base offset )
```

-->

```
Screen # 11
( @EM !EM                      13:05 10/03/85 )
```

```
( em_daddr --- n )
: @EM      4000 UM/MOD 0      ( pa_offs log_pa phys_pa )
          $EM_PID @ em_map ABORT" @EM mapping error"
          $EM_FRAME @ SWAP !L ;
```

```
( n em_daddr --- )
: !EM      4000 UM/MOD 0      ( pa_offs log_pa phys_pa )
          $EM_PID @ em_map ABORT"!EM mapping error"
          $EM_FRAME @ SWAP !L ;
```

```
DECIMAL CR CR .( EMM management routines loaded. )
CR ?WORKSPACE U. .( bytes left in dictionary. ) CR CR
```

End Listing

Product Information

Dr. Dobb's Journal of Software Tools

August 1986 #118

Expiration Date: November 30, 1986

Name _____ Title _____

Company _____ Phone _____

Address _____

City/State/Zip _____

Please circle one letter in each category:

I. My work is performed:

- A. for in-house use only.
- B. for other companies.
- C. for end users/retailers.
- D. in none of the above areas.

II. My primary job function:

- A. Software Project Mgmt./Spvr.
- B. Hardware Project Mgmt./Spvr.
- C. Computer Consultant
- D. Corporate Management
- E. Other

III. My company department performs:

- A. software development.
- B. computer system integration.
- C. computer manufacturing.
- D. computer consulting.
- E. computer research.
- F. none of the above.

IV. This inquiry is for:

- A. a purchase within 1 month.
- B. a purchase within 1 to 6 months.
- C. product information only.

V. Corporate Purchase Authority:

- A. Final Decision-maker
- B. Approve/Recommend
- C. No Influence

VI. Personal Computer Users at my Jobsite:

- A. 10,000 or more
- B. 500 to 9,999
- C. 100 to 499
- D. 10 to 99
- E. less than 10

VII. On average, I advise others about computers:

- A. more than once per day.
- B. once per day.
- C. once per week.
- D. less than once per week.

VIII. In my job function, I:

- A. design software and/or write code.
- B. design software.
- C. write code.
- D. don't design software or write code.

A Reader Service number appears on each advertisement. Circle the corresponding numbers below for more info.

001	002	003	004	005	006	007	008	009	010
011	012	013	014	015	016	017	018	019	020
021	022	023	024	025	026	027	028	029	030
031	032	033	034	035	036	037	038	039	040
041	042	043	044	045	046	047	048	049	050
051	052	053	054	055	056	057	058	059	060
061	062	063	064	065	066	067	068	069	070
071	072	073	074	075	076	077	078	079	080
081	082	083	084	085	086	087	088	089	090
091	092	093	094	095	096	097	098	099	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120
121	122	123	124	125	126	127	128	129	130
131	132	133	134	135	136	137	138	139	140
141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170
171	172	173	174	175	176	177	178	179	180
181	182	183	184	185	186	187	188	189	190
191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220
221	222	223	224	225	226	227	228	229	230
231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260
261	262	263	264	265	266	267	268	269	270
271	272	273	274	275	276	277	278	279	280
281	282	283	284	285	286	287	288	289	290
291	292	293	294	295	296	297	298	299	300

Circle 999 to start a 12 month subscription at the price of \$29.97

Free!

Postage Paid!

Product Information

Free!

Thank You!
Dr. Dobb's greatly appreciates your responses to questions I through VIII.

BUSINESS REPLY MAIL

First Class Permit #217, Clinton, Iowa

POSTAGE WILL BE PAID BY ADDRESSEE

Dr. Dobb's Journal of

Software Tools

P.O. Box 2157

Clinton, Iowa 52735-2157

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES





NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

First Class Permit #217, Clinton, Iowa

POSTAGE WILL BE PAID BY ADDRESSEE

Dr. Dobb's Journal of

Software Tools

P.O. Box 2157

Clinton, Iowa 52735-2157



Thank You!

**Dr. Dobb's greatly appreciates your
responses to questions I through VIII.**

A Reader Service number appears on each advertisement. Circle the corresponding numbers below for more info.

01 002 003 004 005 006 007 008 009
10 011 012 013 014 015 016 017 018
19 020 021 022 023 024 025 026 027
28 029 030 031 032 033 034 035 036
37 038 039 040 041 042 043 044 045
46 047 048 049 050 051 052 053 054
55 056 057 058 059 060 061 062 063
64 065 066 067 068 069 070 071 072
73 074 075 076 077 078 079 080 081
82 083 084 085 086 087 088 089 090
91 092 093 094 095 096 097 098 099
100 101 102 103 104 105 106 107 108
109 110 111 112 113 114 115 116 117
118 119 120 121 122 123 124 125 126
127 128 129 130 131 132 133 134 135
136 137 138 139 140 141 142 143 144
145 146 147 148 149 150 151 152 153
154 155 156 157 158 159 160 161 162
163 164 165 166 167 168 169 170 171
172 173 174 175 176 177 178 179 180
181 182 183 184 185 186 187 188 189
190 191 192 193 194 195 196 197 198
199 200 201 202 203 204 205 206 207
208 209 210 211 212 213 214 215 216
217 218 219 220 221 222 223 224 225
226 227 228 229 230 231 232 233 234
235 236 237 238 239 240 241 242 243
244 245 246 247 248 249 250 251 252
253 254 255 256 257 258 259 260 261
262 263 264 265 266 267 268 269 270
271 272 273 274 275 276 277 278 279
280 281 282 283 284 285 286 287 288
289 290 291 292 293 294 295 296 297
298 299 999

Circle 999 to start a 12 month subscription at the price of \$29.97

Dr. Dobb's Journal of Software Tools

August 1986 #118

Expiration Date: November 30, 1986

Name _____ Title _____

Company _____ Phone _____

Address _____

City / State / Zip _____

Please circle one letter in each category:

I. My work is performed:

- A. for in-house use only.
- B. for other companies.
- C. for end users/retailers.
- D. in none of the above areas.

V. Corporate Purchase Authority:

- A. Final Decision-maker
- B. Approve/Recommend
- C. No Influence

II. My primary job function:

- A. Software Project Mgmt/Spvr
- B. Hardware Project Mgmt/Spvr
- C. Computer Consultant
- D. Corporate Management
- E. Other

VI. Personal Computer Users at my Jobsite:

- A. 10,000 or more
- B. 500 to 9,999
- C. 100 to 499
- D. 10 to 99
- E. less than 10

III. My company department performs:

- A. software development.
- B. computer system integration.
- C. computer manufacturing.
- D. computer consulting.
- E. computer research.
- F. none of the above.

VII. On average, I advise others about computers:

- A. more than once per day.
- B. once per day.
- C. once per week.
- D. less than once per week.

IV. This inquiry is for:

- A. a purchase within 1 month.
- B. a purchase within 1 to 6 months.
- C. product information only.

VIII. In my job function, I:

- A. design software and/or write code.
- B. design software.
- C. write code.
- D. don't design software or write code.

**Product
Information**

Free!

Postage Paid!

**Product
Information**

Free!

NEW! See Inside—

An Updated Version
of the **Unix-like Shell**
for **MS-DOS**
and

Dr. Dobb's Listings
Disk #2

including listings from
DDJ's May through
August 1986 issues,
along with
discounted
back issues!

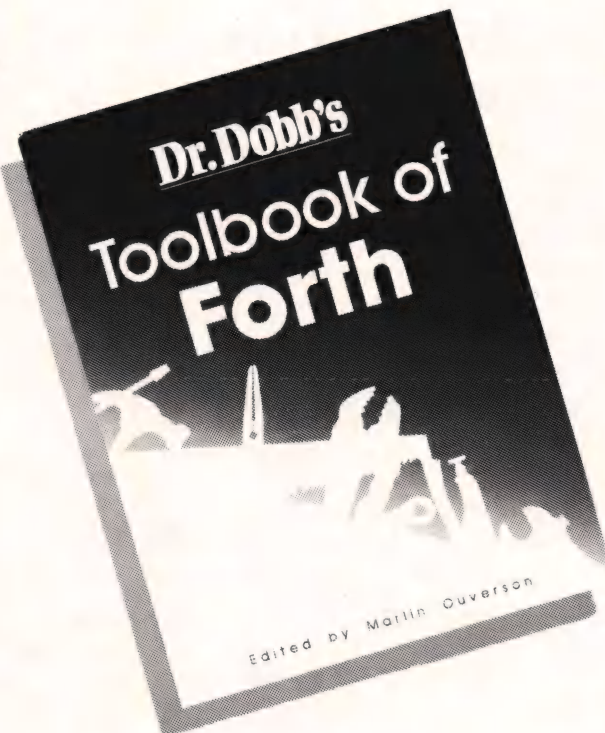
Is Forth the language for you?

It is if you're an advanced user, a systems designer or an applications programmer looking for flexibility, power, speed and minimal program development time.

Forth is also the language for you if you're a beginner who wants to learn more about your computer than you can with a conventional "teaching" language. Whatever your level of expertise, Forth may be for you simply because it is a truly interesting language.

Now, *Dr. Dobb's Journal of Software Tools*, presents Dr. Dobb's Toolbook of Forth.

This comprehensive collection of useful Forth programs and tutorials contains expanded and revised versions of *DDJ's* best Forth articles, along with new Forth material. In addition to the practical code and tutorials, you'll glean important insights about the potential of this increasingly popular language from the many in-depth discussion of advanced Forth topics.



You'll find sections on:

Forth—the Language, including "The Forth Philosophy," "Teaching Forth as a First Language" and "Forth-83 and Vocabularies"

Forth Programs, including "GO in Forth," "Elements of a Forth Data-Base Design," "The Forth Sort," "SEND & RECV," "Interface for a Mouse," "Relocating Loader in Forth," "Forth Decompiler," "Screen-Oriented Editor Revisited," "Evolution of a Video Editor," "H-19 Screen Editor" and "The Conference Tree"

Mathematics in FORTH including "Series Expansion in Forth," "FORTH Floating-Point Package," "Signed Integer Division"

Modifications/Extensions, including "A Proposal for Strings in Forth," "Non-Deterministic Control Words," "Some Forth Coding Standards," "Towards a More Writable Forth Syntax"

Implementing FORTH, including "Forth and the Motorola 68000," "A 68000 Forth Assembler," "A Forth Assembler for the 6502," "Z8000 Forth"

Dr. Dobb's Toolbook of Forth

You'll also find Appendices that will help you convert fig-Forth to Forth-83, and tell you how to stay up-to-date on the latest developments and refinements of this popular language.

The screens in this book are also available on disk as Ascii files. Receive *Dr. Dobb's Toolbook of Forth*, along with the software on disk, together for only \$39.95.

Dr. Dobb's Toolbook of Forth
Item #030 \$22.95

Toolbook of Forth with Disk
Item #031 \$39.95

Please specify MS/PC DOS, Apple II, Macintosh, or CP/M. For CP/M disks, specify Osborne or 8" SS/SD.

To Order:

To order any of Dr. Dobb's products, return the order form at the end of this catalog, or

CALL TOLL FREE
1-800-528-6050
EXT. 4001

and refer to product item number, title, and disk format.
For customer service questions,

CALL M&T PUBLISHING, INC.
415-366-3600 EXT. 216



Dr. Dobb's Catalog

Programs by the Pound

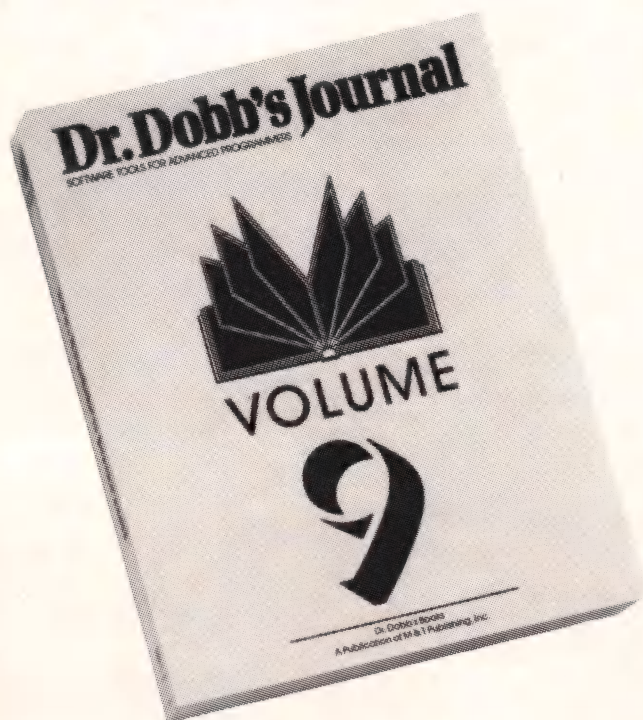
Announcing: Dr. Dobb's Bound Volume 9

Over 1000 pages of listings and text. The entire 1984 editorial contents of *Dr. Dobb's Journal of Software Tools*.

Bound Volume 9: 1984 Item #020B

Shaping things to come. In 1984 new Editor-in-chief Mike Swaine brought his interests in advanced technology to *Dr. Dobb's Journal*. We presented the concepts behind Prolog and published an expert system for weather prediction. We learned Modula-2, and taught Forth to talk to a 68000, to MS DOS, and to the people of China. We examined a new language implementa-

tion called Turbo Pascal and extended implementations of C with a pre-processor, a library, Tony Skjellum's tricks, and Allen Holub's Grep. We published two powerful encryption systems, telecommunications protocols, floating-point benchmark results, and an issue devoted to the internals of Unix. And Ray Duncan, Bob Blum, and Dave Cortesi were on hand with their fascinating columns.



Bound Volume 1: 1976 Item #013

The working notes of a technological revolution. Programmers from Defense laboratory systems analysts to kitchen-table entrepreneurs worked for the intrinsic rewards to put development software on the brand-new invention, the microcomputer. Before there was an Apple, *Dr. Dobb's Journal of Tiny Basic Calisthenics and Orthodontia* (subtitle: Running Light without Overbyte) was founded to put a programming language on the machine, and became both chronicler and instrument of the revolution. In this first-year volume: Tiny Basic, the first word on CP/M, notes on building an IMSAI, floating-point and timer routines.

Bound Volume 2: 1977 Item #014

Running light without overbyte. By year two, *Dr. Dobb's* formula was concocted: tough questions and serious technical issues handled with enthusiasm, and wit, scant reverence for the accepted answers. Source code. Tools for programmers. Respect for tight programming. *Dr. Dobb's Journal* readers shared insights on warping the Intel 8080 into a computer CPU, and *Dr. Dobb's* published a complete operating system for the chip. A motley crop of computers and software products were popping up, and *Dr. Dobb's* investigated: the Heath H-8, the KIM-1, the Alpha Micro, MITS Basic, Poly Basic,

and Lawrence Livermore Labs Basic. *Dr. Dobb's* introduced Pilot for microcomputers and published tips on doing string handling, high-speed I/O, and turtle graphics in limited memory.

Bound Volume 3: 1978 Item #015

The roots of the Silicon Valley growth. In 1978 Steve Wozniak and other programmers were publishing in *Dr. Dobb's Journal* code that would help them grow multi-million-dollar computer companies. The proposed S-100 bus standard was hashed out in *Dr. Dobb's* pages. *Dr. Dobb's* contributors began to speak more in terms of technique than of specific implementations as the industry began to diversify. Languages covered in depth included SAM76, Pilot, Pascal, and Lisp.

To Order:

To order any of Dr. Dobb's products, return the order form at the end of this catalog, or

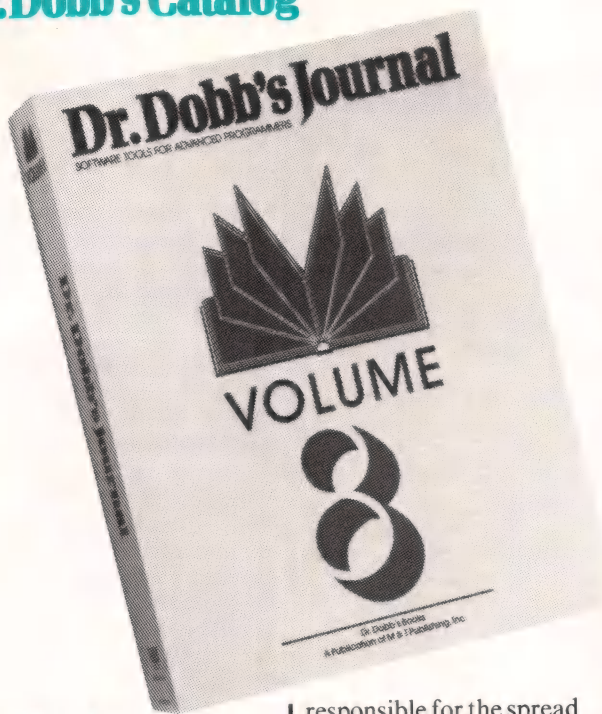
CALL TOLL FREE
1-800-528-6050
EXT. 4001

and refer to product item number, title, and disk format. For customer service questions,

CALL M&T
PUBLISHING, INC.
415-366-3600 EXT. 216



Dr. Dobb's Catalog



Bound Volume 4: 1979

Item #016

In the midst of the Gold Rush. Three years before IBM would release its PC, a thriving, rough-and-tumble personal computer industry existed. Fortunes had been made and lost, the effective power of the machine multiplied a hundredfold. By 1979 some stability had even emerged; one could speak of the processors that had proven longevity as micro-computer CPUs; the 8080, the Z80, the 6800, and the 6502. *Dr. Dobb's Journal* focused on the best ways to use these processors, with algorithms, tips, and code for 8- to 16-bit conversion, pseudo-random number generation, micro-to-mainframe connections, telecommunications, and networking. And lots of useful code.

Bound Volume 5: 1980

Item #017

The preeminence of CP/M and the rise of C. More than any other magazine, *Dr. Dobb's Journal* was

responsible for the spread of CP/M and C on micro-computers. Both of those movements began in 1980. *Dr. Dobb's* all-CP/M issue, including Gary Kildall's history of CP/M, sold out within weeks of publication. This was the year of Ron Cain's original Small C compiler, of a CP/M-oriented C interpreter, CP/M-to-UCSD Pascal file conversion techniques, and a greater concern in *Dr. Dobb's* with software portability.

Bound Volume 6: 1981

Item #018

The first of Forth. 1981 saw *Dr. Dobb's* first all-Forth issue (now sold out), along with an emphasis on CP/M, C, telecommunications, and new languages. David Cortesi began "Dr. Dobb's Clinic," one of the magazine's most popular features. Highlights included information on PCNET, the Conference Tree, the Electronic Phone Book, Tiny Basic for the 6809, writing your own compiler, and a systems programming language.

Bound Volume 7: 1982

Item #019

Legitimacy. By 1982 IBM had become a player in the personal computer game and was changing the rules. New microprocessors arrived, the first designed specifically to serve as personal computer CPUs. In *Dr. Dobb's Journal* Dave Cortesi published the first serious comparison of MS DOS and CP/M-86. *Dr. Dobb's* started two new columns: the CP/M Exchange, as a rearguard maneuver to ensure that good tools for CP/M programmers would continue to be developed and circulated, and the 16-Bit Software Toolbox to investigate the 8088/86 and other new microprocessors. We published code for the 68000 and Z8000 processors, and looked ahead, in a provocative essay, to fifth-generation computers.

Bound Volume 8: 1983

Item #020

Power Tools. Personal computers were proving themselves to be true professional software development tools by 1983, the year in which Jim Hendrix completed his "canonical" version of Small C in *Dr. Dobb's Journal*. *Dr. Dobb's* published more 68000 and 8088 code, and as the memory limitations relaxed, the magazine's commitment to tight code let it shoehorn impossibly large systems into memory. Small C was just one of the major software products published in their entirety in *Dr. Dobb's* pages that year; there were

Ed Ream's RED screen editor and a version of the Ada language called Augusta.

Buy the complete set and save 15%

If you buy all nine volumes, covering the entire editorial content of *Dr. Dobb's Journal of Software Tools* from the first issue in 1976 through 1984, you pay just \$225. That's a 15% discount and over \$40 off the combined individual prices. To order the complete set of Bound Volumes 1 through 9, ask for item #020C.

Vol. 1	Item #013	\$27.75
Vol. 2	Item #014	\$27.75
Vol. 3	Item #015	\$27.75
Vol. 4	Item #016	\$27.75
Vol. 5	Item #017	\$27.75
Vol. 6	Item #018	\$27.75
Vol. 7	Item #019	\$30.75
Vol. 8	Item #020	\$31.75
Vol. 9	Item #020B	\$35.75

All 9 volumes
Item #020C \$225.00

To Order:

To order any of *Dr. Dobb's* products, return the order form at the end of this catalog, or

CALL TOLL FREE
1-800-526-8050
EXT. 4001

and refer to product item number, title, and disk format.

For customer service questions,

CALL M&T

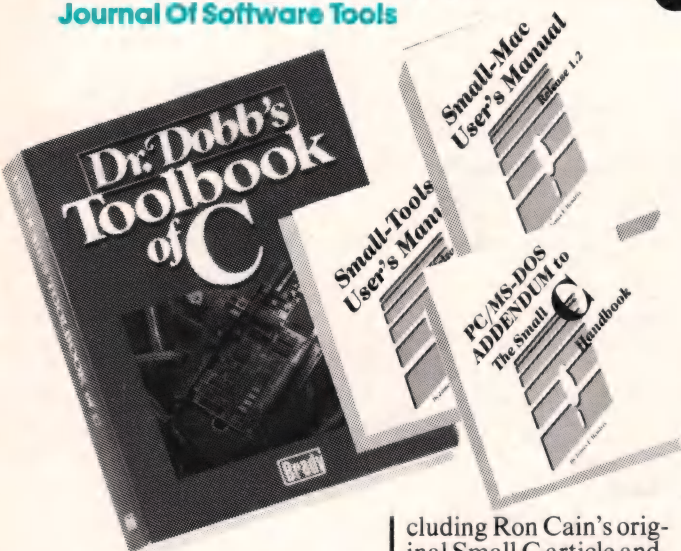
PUBLISHING, INC.

415-366-3600 EXT. 216



A collection of powerful tools
for C software developers,
including books, software on
disk, and reference materials
from the publisher of *Dr. Dobb's
Journal Of Software Tools*

Dr. Dobb's Complete C Toolbox



From M & T Publishing
and Brady
Communications . . .

Dr. Dobb's Toolbook of C Item #005

The **Toolbook** contains
over 700 pages of C mater-
ial, including articles by
such C experts as Kern-
ighan and Ritchie, Cain
and Hendrix, Skjellum
and Holub. The level is so-
phisticated and pragmatic,
appropriate for pro-
fessional C programmers.

The most valuable part
of the **Toolbook** to many
will be the hundreds of
pages of useful C source
code, including a complete
compiler, an assembler,
and text-processing util-
ities. The accompanying
text explains, in the pro-
grammers' own words,
why they did what they
did.

*Dr. Dobb's Journal of
Software Tools* introduced
a generation of personal
computer programmers to
the C programming lan-
guage, and all the best C
articles and code published
in *Dr. Dobb's* over the
years is included and up-
dated in the **Toolbook**, in

cluding Ron Cain's orig-
inal Small C article and
articles from sold-out
issues. But the **Toolbook**
also includes material
never before published, in-
cluding Jim Hendrix's
complete macro assembler
in C.

Dr Dobb's offers the
Toolbook in a special hard-
bound edition for just
\$29.95.

You'll find:
Jim Hendrix's famous
Small C Compiler and **New
Library for Small C** (both
also available on disk),
**NEW: Hendrix's Small
Mac: An Assembler for
Small C** and **Small Tools:
Programs for Text Pro-
cessing** (both also
available on disk), All of
Tony Skjellum's **C Pro-
grammer's Notebook**
columns distilled by Tony
into one thought-
provoking chapter.

Dr. Dobb's C Software Tools on Disk

To complement the **Tool-
book**, *Dr. Dobb's* also
offers the following pro-
grams on disk. Full C
source code and documen-
tation is included. Except
where indicated, both CP/
M and MS/PC DOS ver-
sions are available.

Also from
M & T Publishing and
Brady Communications . . .

The Small C Handbook

Item #006 or #006A

Jim Hendrix's **Small-C
Handbook** is the reference
book on his Small-C com-
piler. In addition to de-
scribing the operation of
the compiler, the book
contains complete source
listings to the compiler and
its library of arithmetic
and logical routines.

Hendrix has ported the
compiler to the MS/PC
DOS environment since
the **Handbook** was printed,
and the **Handbook** plus his
**MS/PC DOS Handbook
Addendum**, Item #006A,
is \$22.95.

A perfect companion to
the Hendrix **Small-C com-
piler** offered by *Dr. Dobb's*
on disk, the **Handbook**
even tells how to use the
compiler to generate a new
version of itself.

While both the **Hand-
book** and the **Toolbook**
provide documentation for
the Small-C compiler, the
Handbook contains a more
detailed discussion and is
available with an adden-
dum for the MS/PC DOS
version.

The **Handbook**, Item
#006, is just \$17.95. Jim

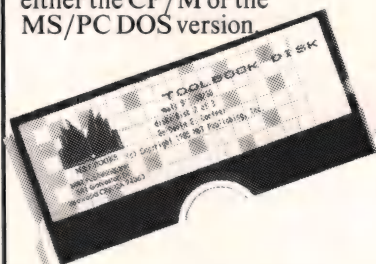
Small-C Compiler Item #007

Jim Hendrix's Small-C
Compiler is the most popu-
lar piece of software ever
published in *Dr. dobb's* 10-
year history. Like a home-
study course in compiler
design, the **Small-C Com-
piler** and **Small-C Hand-
book** provide everything
you need but the computer

for learning how compilers
are constructed, and for
learning C at its most
fundamental level.

While both the **Hand-
book** and the **Toolbook**
provide documentation for
the Small-C compiler, the
Handbook contains a more
detailed discussion and is
available with an adden-
dum for the MS/PC DOS
version. The **MC/PC DOS
Small-C Handbook Ad-
dendum** is recommended in
addition to the **Handbook**
for MS DOS or PC DOS
users.

The **Small-C compiler** is
available for \$19.95 in
either the CP/M or the
MS/PC DOS version.



To Order:

To order any of *Dr.
Dobb's* products, return
the order form at the end
of this catalog, or

**CALL TOLL FREE
1-800-528-6050
EXT. 4001**

and refer to product
item number, title, and
disk format.
For customer service
questions,

**CALL M&T
PUBLISHING, INC.
415-366-3600 EXT. 216**



Dr. Dobb's Complete C Toolbox

Small-Mac: An Assembler for Small-C Item #012A

Small-Mac is an assembler designed to stress simplicity, portability, adaptability, and educational value. The package features a simplified macro facility, C language expression operators, object file visibility, descriptive error messages, and an externally-defined machine instruction table. You get the macro assembler, linkage editor, load-and-go loader, library manager, CPU configuration utility, and a utility to dump relocatable files.

Small-Mac is available with documentation for \$29.95. For CP/M systems only.

Small-Tools: Programs for Text Processing

A package of programs performing specific, modular operations on text files, including: editing; formatting; sorting; merging; listing; printing; searching; changing; transliterating; copying; concatenating; encrypting and decrypting; replacing spaces with tabs and tabs with spaces; counting characters, words, or lines; and selecting printer fonts. Supplied in source code form so you can select and adapt these tools to your own purposes.

Small-Tools is available with documentation for \$29.95. For CP/M or MS/PC DOS systems.

Special Packages—20% Off

Now for almost 20% off the combined individual

product prices, you can order a complete set of *Dr. Dobb's C programming tools* for your CP/M or MS/PC DOS system.

C Package for CP/M Item #005A

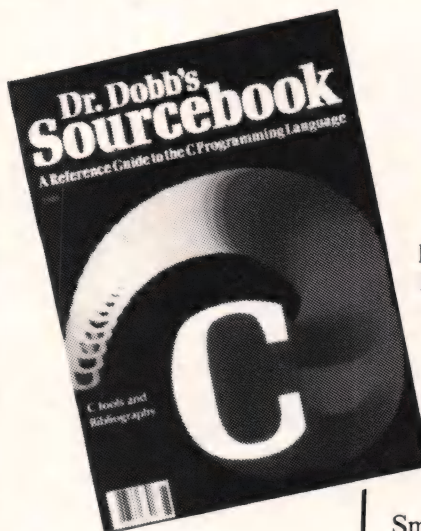
Ordered individually, these items would cost about \$120. If you order the CP/MC package, you'll get *Dr. Dobb's Toolbook*, the *Small-C Handbook*, the *Small-C Compiler* on disk, the *Small-Mac* assembler on disk with documentation in the *Small-Mac Manual*, the *Small-Tools* text-processing programs on disk with documentation in the *Small-Tools Manual*, all for just \$99.95.

C Package for MS/PC DOS Item #005B

These items would cost over \$100 if purchased individually. If you order the MS/PC DOS C package, you'll get *Dr. Dobb's Toolbook*, the *Small-C Handbook* with the *MS/PC DOS Addendum*, the *Small-C Compiler* on disk, the *Small-Tools* text-processing programs on disk with documentation in the *Small-Tools Manual*, all for just \$82.95.

Dr. Dobb's Sourcebook: A Reference Guide to the C Programming Language Item #004

Products and services for C programmers are appearing at so rapid a rate that it's all but impossible to keep up with them. *Dr. Dobb's* presents this handy



guide to the who, what, when, where, and why of C. A comprehensive reference manual to new information, products, and services, the *Sourcebook* contains:

- A bibliography of over 300 articles and books on the C language;
- A descriptive list of products for C programmers: compilers, editors, interpreters, and utilities;
- A list of C-related services: classes, seminars, and on-line services.

The *Sourcebook* is just \$7.95. *Dr. Dobb's Sourcebook: A Reference Guide to the C Programming Language*

Item #004 \$ 7.95

Dr. Dobb's Toolbook for C
Item #005 \$29.95

The *Small-C Handbook*
Item #006 \$17.95

The *Small-C Handbook and MS/PC-DOS Addendum*
Item #006A \$22.95

Small-C Compiler disk
Item #007 \$19.95

MS/PC DOS *Small-C Handbook Addendum*
Item #008 \$4.95

Small Tools: Programs for Text Processing disk & manual
Item #010A \$29.95

Small Mac: An Assembler for Small-C disk (For CP/M only) & manual.
Item #012A \$29.95

CP/M C Package
Item #005A \$99.95

MS/PC DOS C Package
Item #005B \$82.95

For CP/M disks, please specify one of the following formats: Apple, Osborne, Kaypro, Zenith Z-100 DS/DD, 8" SS/SD.

To Order:

To order any of *Dr. Dobb's products*, return the order form at the end of this catalog, or

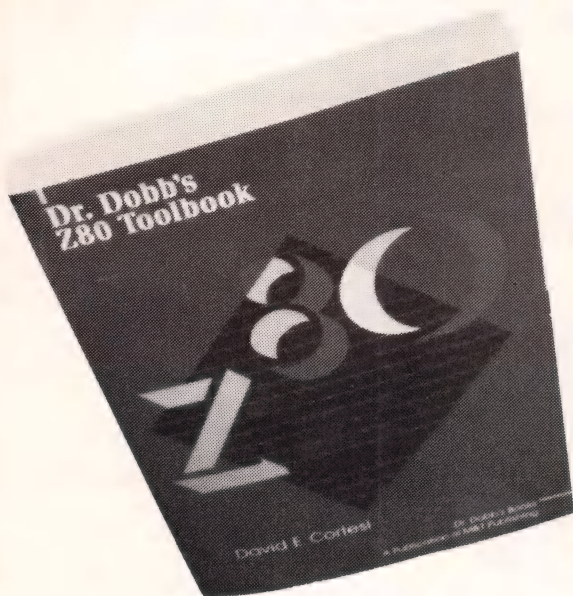
**CALL TOLL FREE
1-800-528-6050
EXT. 4001**

and refer to product item number, title, and disk format. For customer service questions,

**CALL M&T
PUBLISHING, INC.
415-366-3600 EXT. 216**



Dr. Dobb's Z80 Toolbook



David E. Cortesi longtime Dr. Dobb's columnist and author of *Inside CP/M* brings you —

Dr. Dobb's Z80 Toolbook

Here's all you need to write your own Z80 assembly language programs for only \$25!

Do you use CP/M? Do you feel as if the only part of the computer industry that has **not** abandoned you is your own Z80 computer? It keeps on working, but when you need programs for it, you have to write them yourself. When you do, you quickly find that while Pascal or BASIC is okay for some things, there is often no substitute for the speed, small size, and flexibility of an assembly language program.

Dr. Dobb's Z80 Toolbook puts the power of assembly language in the hands of anyone who's done a little programming. You'll find:

****A method of designing programs and coding them in assembly language.** Cortesi will take you on a walk through the initial specifications, designing an algorithm and writing the code. He demonstrates this method in the construction of several complete, useful programs.

****A complete, integrated toolkit of subroutines** for arithmetic, for string-handling, and for total control of the CP/M file system. They bring the ease and power of a compiler's runtime library to your assembly language work, without a compiler's size and sluggish code.

Best of all, every line of the toolkit's source code is there for you to read, and every module's operation is explained with the clarity and good humor for which Dave Cortesi's writing is known.

Order the Z80 Software on Disk! Save Yourself the Frustration of File Entry

All the software in **Dr. Dobb's Z80 Toolbook**—the programs

plus the entire toolkit, both as source code and as object modules for both CP/M 2.2 and CP/M Plus—is yours on disk. (A Z80 microprocessor and a Digital Research International RMAC assembler or equivalent are required.)

Receive Dr. Dobb's Toolbook for Z80, along with the software on disk, together for only \$40!

Dr. Dobb's Toolbook for Z80
Item #022 \$25

Dr. Dobb's Toolbook for Z80, together with software on disk. Please specify one of the following formats: 8" SS/SD; Apple; Osborne; Kaypro.

Item #022A \$40

Most of the programs are included in the book; however, the disk is necessary for complete listings

To Order:

To order any of Dr. Dobb's products, return the order form at the end of this catalog, or

**CALL TOLL FREE
1-800-528-6050
EXT. 4001**

and refer to product item number, title, and disk format.

For customer service questions,

**CALL M&T
PUBLISHING, INC.
415-366-3600 EXT. 216**



A UNIX-like Shell and Utility Package for MS-DOS



A UNIX-like Shell for MS-DOS, V.2 and A UNIX-like Utility Package for MS-DOS by Dr. Dobb's C-Chest Columnist, Allen Holub
Only \$29.95 each!
If you are a registered user or have already purchased version 1 of The Shell from DDJ, you can receive the upgrade for only \$6!

THE SHELL Version 2.0

An MS-DOS implementation of the most often used parts of the UNIX C Shell. This package includes an executable version of the shell, along with the **complete C source code and full documentation**. Supported features are: **Editing** Command-line editing with the cursors is supported. The line is visible as you edit it. **Aliases** Can be used to change the names of commands or as very fast, memory-resident, batch files. Nested aliases are supported in version 2. **History** You can execute previous commands. The command can be edited before being executed. Version 2 supports embedded history requests (bar; !!>foo).

Redirection and Pipes

```
< > >>
    >& >>&
```

Pipe temporary files can be put on a RAM disk.

Unix-like Command

Syntax: / can be used to separate directory names (\ can now be used as well).

A 2048-byte command line is supported. Command-line wild card expansion. Multiple commands on a line.

DOS-compatible prompt support \$d \$t \$- \$e \$h \$n \$q \$\$\$ \$%

C-Shell Based Shell

Scripts (batch files) Shell Variables are macros that can be used on the command line. Version 2 supports arithmetic manipulation of shell variables using the @ command. The following C operators are supported:
 () + - * / %
 < = > = < > !=
 == ! && || =

A batch file can call another batch file like a subroutine. Control is passed to the second file and then back to the first when the second is finished. Batch files can return values to the calling file using the exit and \$status mechanisms.

A powerful, interpretive, programming language, based on the UNIX C Shell, is now supported, including:

```
if/then/else
while
foreach
switch/case
break
continue
```

All commands can be nested.

/util

A UNIX-like Utility Package for MS-DOS This collection of utility programs for MS-DOS includes updates of the highly acclaimed Dr. Dobb's articles *Grep: A UNIX-Like Generalized Regular Expression Processor* *Ls* from Dr. Dobb's C Chest column,

and *Getargs* from DDJ's C Chest.

Source code is included and all programs (and most of the utility subroutines) are fully documented in a UNIX-style manual. You'll find executable versions of:

cat A file concatenation and viewing program
cp A file copy utility
date Prints the current time and date
du Prints amount of space available and used on disk
echo Echoes its arguments to standard output
grep Searches for a pattern defined by a regular expression
Ls Gets a sorted directory
mkdir Creates a directory
mv Renames a file or directory. Moves files to another directory
p Prints a file, one page at a time
pause Prints a message and waits for a response
printenv Prints all the environment variables
rm Deletes one or more files
rmdir Deletes one or more directories
sub Text substitution utility. Replaces all matches of a regular expression with another string.
chmod change all file attributes (write permission, hidden, system, archive bit).

ORDER THE SHELL AND /UTIL TOGETHER FOR ONLY \$50!
SAVE OVER 15%

The Shell runs on IBM PC's and compatibles

The Shell	
Item #160	\$29.95
Shell Upgrade Disk for owner of the Shell v.1	
Item #160A	\$ 6.00
/util	
Item #161	\$29.95
Shell/util Package	
Item #162	\$50.00

To Order:

To order any of Dr. Dobb's products, return the order form at the end of this catalog, or

CALL TOLL FREE
1-800-528-6050
EXT. 4001

and refer to product item number, title, and disk format.

For customer service questions,

CALL M&T PUBLISHING, INC.
415-366-3600 EXT. 216



NEW!



Dr. Dobb's Listings

You can save time entering Dr. Dobb's valuable code listings! Selected listings from *this August 1986 issue*, along with listings from all previous 1986 DDJ issues, are yours on disk!

Now, as a useful adjunct to the magazine, DDJ offers the additional value and convenience of selected listings from each 1986 issue, on disk. The first two of three Dr. Dobb's 1986 Listings disks, featuring a collection of listings from the DDJ January 1986 issue through *this August 1986 issue*, are now available.

And, in case you missed a single valuable issue of Dr. Dobb's Journal of Software Tools, while the supply lasts you can also receive any available 1986 back issue for only \$2.50 each when you purchase a DDJ Listings disk. That's a savings of 50% off the regular back issues price!

Dr. Dobb's Listings Disk #1/86

January-April 1986 You'll find listings from the following articles, among others:

From issue #111 January 1986

****A Simple OS for Real-time Applications;** 68000 assembly language techniques for an operating system kernel

by DDJ editor Nick Turner

****Exec calls and Fortran;** a technique allowing execution of a user or system task from a user program from DDJ's 16-Bit Software Toolbox, by Robert Sypek

****32-bit Square Roots;** An 8086 assembly-language routine for 32-bit square roots by Michael Barr (Sorry, back issue #111 is sold out)

From issue #112 February 1986

****Fast Integer Powers for Pascal;** An implementation of the fastest-known algorithm for the computation of integer powers by Dennis E. Hamilton

****Data Abstraction with Modula-2;** Construction of a priority queue in Modula-2 by Bill Walker and Stephen Alexander

****Learning Ada on a Micro;** A draw poker program in Ada by Do-While Jones

****Fast IBM PC graphics routines** from DDJ's 16-Bit Software Toolbox, by Dan Rollins (Sorry, back issue #112 is sold out)

From issue #113 March 1986

****Recursive Bose-Nelson Sort;** An alternative to Joe Celko's September 1985 sort routine by R.J. Wissbaum

****A Variable-Metric Minimizer;** A C program for minimizing arbitrary functions by Joe Marasco

****Concurrency and Turbo Pascal;** An approach to implementing

coroutine in Pascal by Ernest Bergmann

****Speeding MS DOS Disk Access;** Programs to test disk-access speed by Greg Weissman

****Square Roots on the NS32000;** Comparable square root routines in C and assembly language for National Semiconductor's 32000 family by Richard Campbell

From issue #114 April 1986

****Boca Raton Inference Engine;** Lisp, Prolog, and Expert-2 techniques and code by Robert Brown

Dr. Dobb's Listings

Disk #2/86

May-August 1986 You'll find listings from the following articles, among others.

From issue #115 May 1986

****Simple plots with the Enhanced Graphics Adapter** by Nabajyoti Barkakati

****The Cryptographer's Toolbox** by Fred A. Scacchitti

From issue #116 June 1986

****Structured Programming;** Overloading Procedures, Exporting Opaque Types, Data Hiding by Namir Shamma

****Compuserve B Protocol** by Steve Wilhite

From issue #117 July 1986

****Structured Programming;** Tiny Tools, Array-Defining Words by Michael Ham

From issue #118 August 1986

****Structured Programming;** Generic Routines in Ada and Modula-2, Extended for Loop by Namir Shamma

Dr. Dobb's Listings Disk #1/86

Item #170 \$14.95

Dr. Dobb's Listings Disk #2/86

Item #171 \$14.95

Please specify MS/DOS, Macintosh, or CP/M. For CP/M disks, please specify one of the following formats: Apple, Osborne, Kaypro, Zenith Z-100 DS/DD, 8" SS/SD.

To order any DDJ 1986 back issue at the discounted price of only \$2.50 each with your Dr. Dobb's Listings Disk, please specify the issue date and number on the order form. January Issue #111 and February Issue #112 are sold out.

To Order:

To order any of Dr. Dobb's products, return the order form at the end of this catalog, or

**CALL TOLL FREE
1-800-528-6050
EXT. 4001**

and refer to product item number, title, and disk format.

For customer service questions,

**CALL M&T
PUBLISHING, INC.
415-366-3600 EXT. 216**



Dr.Dobb's Catalog

Order Form

ORDER NOW

NAME _____

(Please use street address, not P.O. Box)

ADDRESS _____

CITY _____

STATE _____

ZIP _____

DAY PHONE _____

For disk orders, please indicate format. Refer to ad for standard format availability for each product.

Special formats available for additional \$10 each.

☐ MS/DOS

☐ Macintosh

☐ CP/M

_____ Osborne

☐ Apple II

_____ Zenith Z-100 DS/DD

_____ Kaypro

_____ Apple

_____ 8" SS/SD

For Faster Service
On Credit Card Orders

CALL TOLL FREE
1-800-528-6050
Ext. 4001

Please Refer To Item #
When Ordering

Or, Fill Out This Postage Paid,
Self-Mailing Order Form
And Return To:

M&T PUBLISHING INC.
501 GALVESTON DRIVE
REDWOOD CITY, CA 94063

QUANTITY	ITEM #	DESCRIPTION	UNIT PRICE	TOTAL PRICE

CA residents must add applicable sale tax on merchandise total _____ %

(CA residents must add sales tax EXCEPT *Dr. Dobb's Sourcebook* #004)

Shipping must be included with order. See rates below.

In U.S. For Bound Volumes, add \$2.25 per book. Add \$8.75 for special C Packages. For other books and disks, add \$1.75 per item.

Outside U.S. For Bound Volumes, add \$5.25 per book surface mail. Add \$18 surface mail for Special C Packages. For other books and disks, add \$3.25 per item surface mail. Foreign airmail rates available on request.

SUB-TOTAL _____

SALES TAX _____

SHIPPING _____

TOTAL ORDER _____

☐ VISA

NAME ON CARD _____

☐ MASTERCARD

ACCOUNT NO. _____

☐ AMERICAN EXPRESS

EXPIRATION DATE _____

☐ CHECK

SIGNATURE _____

(make checks payable
to M&T Publishing)

PROMPT DELIVERY! DEALER INQUIRY WELCOME!



Dr.Dobb's Catalog Order

Please Rush!

Please fold along fold-line and staple or tape closed.



No Postage
Necessary
If Mailed
In The
United States

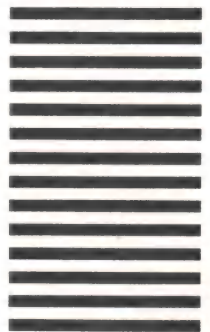
BUSINESS REPLY MAIL

First Class Permit No. 790 Redwood City, CA

Postage Will Be Paid By Addressee

Dr.Dobb's Catalog

501 Galveston Dr.
Redwood City, CA 94063



Please fold along fold-line and staple or tape closed.

LETTERS

Listing One (Text begins on page 10.)

```

Listing One
;DeSmet function isqrt(source):
;...source is a long integer (32 bit)...
;Returns square root of source in ax, using Newton's method: see Scanlon's
;8086 book for similar function (Scanlon's is not sufficiently general, and has
;an error)...
;
;REGISTER USAGE      cx:bx    ...stores copy of 32-bit source throughout...
;                    di        ...'last' estimate of isqrt(source)...
;                    si        ...current estimate of isqrt(source)...
;                    dx:ax     ...used to divide source by di...

cseg
public isqrt_
isqrt_:
    push bp
    mov bp,sp
    mov bx,[bp+4]    ;Store a copy of source in cx:bx;
    mov cx,[bp+6]    ;cx:bx preserved till almostdone...
;-----Start block to determine initial estimate: base estimate on most-----
;-----significant non-zero byte of cx:bx-----
    cmp ch,0        ;Note 46341 covers largest positive
    je test_cl      ;long that most compilers will pass.
    mov di,46341    ;If you use 32-bit unsigned, replace
    jmp load_si     ;with 65535, and if cx>=FFFEH, jump
;-----
    test_cl:        cmp cl,0
    je test_bh      ;out and set result= 65535.
    mov di,7896
    jmp load_si
;-----
    test_bh:        cmp bh,0
    je its_bl
    mov di,181
    jmp load_si
;-----
    its_bl:         mov di,8
    load_si:        mov si,di
;-----End block to determine initial estimate-----

;-----Begin loop to refine the estimate-----
refine:            mov dx,cx    ;Load dx:ax pair with source in prep
    mov ax,bx      ;for divide by di-estimate...
    div di
;-----Block to average quotient and last estimate-----
    shr ax,1       ;We can't just add di,ax then
    adc di,0       ;shr di,1 because sum of di and ax
    shr di,1       ;may exceed 65535...
;-----
    sub si,di      ;Obtain difference betw. old (si)
    jz almostdone  ;and new estimates; if 0, we're
    cmp si,1       ;almost done...Else if diff. is
    je almostdone  ;1 or -1 we're almost done...
    cmp si,-1
    je almostdone
    mov si,di      ;Store current value di in si as
                    ;'old' estimate for next iteration.
    jmp refine
;-----End loop to refine the estimate-----
almostdone:       mov ax,di
    mul di
    sub bx,ax
    sbb cx,dx
    jns done
    dec di
    mov ax,di
    mov sp,bp
    pop bp
    ret

/* Test driver for isqrt()... */
main()
{
    long start,i,NumSqrts;
    printf("ENTER start: ");
    scanf("%d",&start);
    printf("ENTER NumSqrts: ");
    scanf("%d",&NumSqrts);
    printf("isqrt(%d) = %u\n",start,isqrt(start));
    putchar('\007');
    for(i=start;i<start+NumSqrts;i++)isqrt(i); /* Remove this isqrt() to find */
    putchar('\007'); /* time taken by loop itself...*/
}

/* Another short driver: this one better for verifying algorithm */
main()
{
    long source;
    unsigned result;
    printf("ENTER # for sqrt: negative exits\n");
    while (printf("ENTER #: ",scanf("%d",&source)),source>=0){
        result=isqrt(source);
        printf("result= SQRT(%d)= %u\n",source,result);
        printf("result*result= %d\n",((long)result)*((long)result));
        printf("(result+1)*(result+1)= %d\n\n", (result+1L)*(result+1L));
    }
}

```

End Listing One

Listing Two

Listing Two Bit-Shifting Method (Slower)

```

;DeSmet function isqrt(): takes a long (32-bit) integer as an argument, returns
;a short (16-bit) integer square root. Function result returned in ax.
;Modified after 68000 code published in DDJ #109, Nov. 1985, p. 90. Comments
;give roughly analogous 68000 instructions; correspondence with 68000 registers
;is:
;DO = sp:si (initially holds argument)

```

(continued on next page)

Put More UNIX™ in Your C.

Unitools \$99

MAKE, DIFF and GREP

These versatile UNIX-style utilities put power at your fingertips. MAKE, a program administrative tool, is like having an assistant programmer at your side. DIFF compares files and shows you the differences between them. GREP can search one or many files looking for one pattern or a host of patterns.



"Z" \$99

A Powerful "vi"-type Editor:

Similar to the Berkeley "vi" editor, "Z's" commands are flexible, terse, and powerful; macro functions give you unlimited range. Features include "undo," sophisticated search and replace functions, automatic indentation, C-tags, and much, much more.



PC-LINT \$99

Error Checking Utility

A LINT-like utility that analyzes programs and uncovers bugs, quirks and inconsistencies. Detects subtle errors. Supports large and small memory models, has clear error messages and executes quickly. Has lots of options and features that you wouldn't expect at this low price.



SunScreen \$99

Low-priced Screen Utility

This versatile graphics package easily creates and modifies formatted screens, validates fields, supports function keys, color and monochrome cards. With library source SunScreen is \$199.

Compatible with all leading MS/PC-DOS C compilers.

SPECIAL OFFER:
Unitools, "Z,"
PC-LINT and Sun-
Screen All for only

\$349

To order or for information call:

TECWARE
1-800-TEC-WARE

(In NJ call 201-530-6307)



UNIX is a registered TM of Bell Laboratories. MAKE, ZTEC, TM Mass Software Systems, Inc. PC-LINT, TM GIMPLE software, SunScreen, TM SunTec, MS-DOS, TM Microsoft

Circle no. 109 on reader service card.

RED

Full Screen Text Editor

IBM PC, Kaypro, CP/M 80 and CP/M 68K systems.

- RED is fast! RED uses all of your terminal's special functions for best screen response. RED handles files as large as your disk automatically and quickly.
- RED is easy to use for writers or programmers. RED's commands are in plain English.
- RED comes with complete source code in standard C. RED has been ported to mainframes, minis and micros.
- RED comes with a Reference Card and a Reference Manual that provides everything you need to use RED immediately.
- RED is unconditionally guaranteed. If for any reason you are not satisfied with RED your money will be refunded promptly.

RED with manual: \$95**Manual only: \$10**

Call or write today for more information:
Enteleki, Inc.
210 N. Bassett St., Room 101
Madison, WI 53703
Tele. (608) 258-7078

To order:

Either the BDS C compiler or the Aztec CII compiler is required for CP/M80 systems. Digital Research C compiler v1.1 is required for CP/M 68K systems. No compiler is required for IBM or Kaypro systems.

Specify both the machine desired (IBM, Kaypro or CP/M) and the disk format described (8 inch CP/M single density or exact type of 5 1/4 inch disk).

Send a check or money order for \$95 (\$105 U.S. for foreign orders). Sorry, I do NOT accept phone, credit card, or COD orders. Please do not send purchase orders unless a check is included. Your order will be mailed to you within one week.

Dealer inquiries invited.

Circle no. 90 on reader service card.



to



the dBx™ translator

- dBx produces quality C direct from dBASE II or III programs.
- Move dBASE programs to UNIX or other machines.
- Improve program speed and reliability.
- Support multi-user/network applications.
- With power guidebook of conversion hints.
- Includes full screen handler and uses your current C database manager.
- May be used to move existing programs or help dBASE programmers learn C easily.
- For MSDOS, PC DOS, UNIX, XENIX, Macintosh, AMIGA. (Uses ANSL.SYS driver on MSDOS, CURSES under UNIX)
- Priced from \$350, also available from distributors.

dBx is a trademark of **Desktop Ai**

1720 Post Road E., Westport, CT 06880 MCIMAIL • DESKTOPAI
Phone • 203-255-3400 Telex • 6502972226MCI

Circle no. 258 on reader service card.

C-Now that you've learned it, find out how to use it best!

New!

APPLIED C

Edited by Strawberry Software, Watertown, MA

Let experts from Lattice, Greenleaf Software, XOR Corporation, AGS Computers, and Phoenix Software show you—through clear instructions and examples—how to use C to solve your advanced programming problems. They give you full details on the hottest new area in C-programming—microcomputer applications—and trace the application development process from beginning to end. They explain program specification, user interface design, style, portability, and numerous other vital topics, many of them covered for the first time ever. 272 pages, 6 x 9, 30 illustrations, \$37.95

Also of interest:

New!

EMBEDDED PROGRAMMING IN ADA®

By Theodore F. Elbert, University of West Florida, Pensacola, 512 pages, 6 x 9, illustrated, \$45.95

LISP**The Language of Artificial Intelligence**

By A.A. Berk, 168 pages, 6 x 9, 27 line drawings, \$24.95

Mail this coupon now for your FREE 15-day examination copies!

VNR VAN NOSTRAND REINHOLD

Mail Order Service
7625 Empire Drive, Florence, KY 41042

YES! Please send me the book(s) checked below for 15 days' **FREE EXAMINATION**. At the end of that time, I will send the purchase price plus local sales tax and a small shipping/handling charge, or simply return the book(s) and **OWE NOTHING**.

- 28217-6 Applied C \$37.95
— 22350-1 Embedded Programming in ADA® \$45.95
— 20974-6 LISP: The Language of Artificial Intelligence \$24.95

☐ **SAVE MONEY!**—Check here if enclosing payment with order and publisher pays shipping/handling. 15-day return/refund guarantee applies. Local sales tax must be included.

Name _____

Address _____

(No shipment to P.O. box addresses without prepayment.)

City _____

State _____ Zip _____

☐ **Mastercard** ☐ **VISA** ☐ **American Express**—Please charge my credit card. Publisher pays shipping/handling. If I return the book(s) at the end of 15 days, charges will be canceled.

Card # _____ Exp. ____/____

Signature _____

Offer good in U.S.A. and territorial possessions only and subject to credit department approval. Prices subject to change. Prices slightly higher in Canada. D 8121

**Van Nostrand Reinhold**

Circle no. 256 on reader service card.



Thinking about C?

**Stop Thinking-
Start Programming Today!**

SPECIAL INTRODUCTORY OFFER!
C' Prime, **Personal Computing** and C,
Plus Apprentice C.
A \$169 value only **\$99**

NEW FROM MANX AZTEC!

C' Prime ~~\$99~~ **\$79**

Never has C been easier to learn. **Manx Aztec** is now offering a complete C system called **C' Prime** at an exceptionally low price. This powerful system includes a Compiler, Linker, Assembler, Editor, Libraries and Object Librarian. **C PRIME** supports a host of third-party software.

C Apprentice ~~\$49.95~~ **\$39.95**

Learn C quickly with this complete, easy-to-use C language interpreter. Apprentice C includes a complete one-step compiler that executes with lightning speed, an editor, and a run-time system.

NEW FROM ASHTON-TATE!

Personal Computing and C

A detailed, easy-to-understand guide to C programming prepared especially by Ashton-Tate for use with the new Aztec C' Prime. Includes chapters on C programming basics, function libraries, data handling, and advanced features, plus a complete money management demonstration program.

To order or for information call:

TECWARE
1-800-TEC-WARE

(In NJ call 201-530-6307)



UNIX is a registered TM of Bell Laboratories, ©1984 TM Aztec Tech, Inc. MANX AZTEC C' PRIME, Apprentice C TM Manx Software Systems, Inc.

LETTERS

Listing Three (Listing continued, text begins on page 10.)

```

.....
Faster Integer Square Root (16 to 8 bit).For small arguments.
(Exact method, not approximate).

Call with:
DO.W = Unsigned number.

Returns:
DO.W = SQRT(DO.W)

Notes: Result fits in DO.B, but is valid in word.
Takes from 72 (d0=1) to 504 (d0=625) cycles
(including rts).

Algorithm supplied by Motorola.
.....

* Use the theorem that a perfect square is the sum of the first
sqrt(arg) number of odd integers.

*
Cycles
move.w d1,-(sp) (8)
move.w #1,d1 (8)
qsqrt1 addq.w #2,d1 (4)
sub.w d1,d0 (4)
bpl qsqrt1 (10/8)
asr.w #1,d1 (8)
move.w d1,d0 (4)
move.w (sp)+,d1 (12)
done rts (16)

.....

Integer Square Root (16 to 8 bit).
(Exact method, not approximate).

Call with:
DO.W = Unsigned number.

Returns:
DO.L = SQRT(DO.W)

Uses:
D1-D4 as temporaries --
D1 = Error term;
D2 = Running estimate;
D3 = High bracket;
D4 = Loop counter

Notes: Result fits in DO.B, but is valid in word.

Takes from 512 to 592 cycles (including rts).

Instruction times for branch-type instructions
listed as (X/Y) are for (taken/not taken).

.....

*
Cycles
qsqrt movem.w d1-d4,-(sp) (24)
move.w #7,d4 (8) ; Loop count (bits-1 of result).
clr.w d1 (4) ; Error term in D1.
clr.w d2 (4)
sqrt1 add.w d0,d0 (4) ; Get 2 leading bits a time and add
addx.w d1,d1 (4) ; into Error term for interpolation.
add.w d0,d0 (4) ; (Classical method, easy in binary).
addx.w d1,d1 (4) ; Running estimate * 2.
add.w d2,d2 (4)
move.w d2,d3 (4)
add.w d3,d3 (4)
cmp.w d3,d1 (4)
bls.s sqrt2 (10/8) ; New Error term > 2* Running estimate?
addq.w #1,d2 (4) ; Yes, we want a '1' bit then.
addq.w #1,d3 (4) ; Fix up new Error term.
sub.w d3,d1 (4)
sqrt2 dbra d4,sqrt1 (10/14) ; Do all 8 bit-pairs.
move.w d2,d0 (4)
movem.w (sp)+,d1-d4 (28)
rts (16)

.....

Integer Square Root (32 to 16 bit).
(Exact Method, not approximate).

Call with:
DO.L = Unsigned number.

Returns:
DO.L = SQRT(DO.L)

Uses:
D1-D4 as temporaries --
D1 = Error term;
D2 = Running estimate;
D3 = High bracket;
D4 = Loop counter.

Notes: Result fits in DO.W, but is valid in longword

Takes from 1080 to 1236 cycles (including rts.)

Two of the 16 passes are unrolled from the loop so
quicker instructions may be used where there is no
danger of overflow (in the early passes).

Instruction times for branch-type instructions
listed as (X/Y) are for (taken/not taken).

```


COMBINE THE RAW POWER OF FORTH WITH THE CONVENIENCE OF CONVENTIONAL LANGUAGES

HS/ FORTH

Why HS/FORTH? Not for speed alone, although it is twice as fast as other full memory Forths, with near assembly language performance when optimized. Not even because it gives MANY more functions per byte than any other Forth. Not because you can run all DOS commands plus COM and EXE programs from within HS/FORTH. Not because you can single step, trace, decompile & disassemble. Not for the complete syntax checking 8086/8087/80186 assembler & optimizer. Nor for the fast 9 digit software floating point or lightning 18 digit 8087 math pack. Not for the half megabyte LINEAR address space for quick access arrays. Not for complete music, sound effects & graphics support. Nor the efficient string functions. Not for unrivaled disk flexibility — including traditional Forth screens (sectored or in files) or free format files, all with full screen editors. Not even because I/O is as easy, but far more powerful, than even Basic. Just redirect the character input and/or output stream anywhere — display, keyboard, printer or com port, file, or even a memory buffer. You could even transfer control of your entire computer to a terminal thousands of miles away with a simple >COM <COM pair. Even though a few of these reasons might be sufficient, the real reason is that we don't avoid the objections to Forth — WE ELIMINATE THEM!

Public domain products may be cheap; but your time isn't. Don't shortchange yourself. Use the best. Use it now!

HS/FORTH, complete system: \$395. with "FORTH: A Text & Reference" by Kelly and Spies, Prentice-Hall and "The HS/FORTH Supplement" by Kelly and Callahan



Visa

Mastercard



HARVARD SOFTWARES

PO BOX 69
SPRINGBORO, OH 45066
(513) 748-0390

Circle no. 132 on reader service card.

```

*
*****
*
*               Cycles
qlsqr      movem.l  d1-d4,-(sp) (40)
            moveq #13,d4      (4)      ; Loop count (bits-1 of result).
            moveq #0,d1       (4)      ; Error term in D1.
            moveq #0,d2       (4)
lsqr1      add.l  d0,d0      (8)      ; Get 2 leading bits a time and add
            addx.w d1,d1      (4)      ; into Error term for interpolation.
            add.l  d0,d0      (8)      ; (Classical method, easy in binary).
            addx.w d1,d1      (4)
            add.w  d2,d2      (4)      ; Running estimate * 2.
            move.w d2,d3      (4)
            add.w  d3,d3      (4)
            cmp.w  d3,d1      (4)
            bls.s  lsqr2      (10/8)   ; New Error term > 2* Running estimate?
            addq.w #1,d2      (4)      ; Yes, we want a '1' bit then.
            addq.w #1,d3      (4)      ; Fix up new Error term.
            sub.w  d3,d1      (4)
lsqr2      dbra d4,lsqr1     (10/14)   ; Do first 14 bit-pairs.

            add.l  d0,d0      (8)      ; Do 15-th bit-pair.
            addx.w d1,d1      (4)
            add.l  d0,d0      (8)
            addx.l d1,d1      (8)
            add.w  d2,d2      (4)
            move.l d2,d3      (4)
            add.w  d3,d3      (4)
            cmp.l  d3,d1      (6)
            bls.s  lsqr3      (10/8)
            addq.w #1,d2      (4)
            addq.w #1,d3      (4)
            sub.l  d3,d1      (8)

lsqr3      add.l  d0,d0 (8)      ; Do 16-th bit-pair.
            addx.l d1,d1      (8)
            add.l  d0,d0      (8)
            addx.l d1,d1      (8)
            add.w  d2,d2      (4)
            move.l d2,d3      (4)
            add.l  d3,d3      (8)
            cmp.l  d3,d1      (6)
            bls.s  lsqr4      (10/8)
            addq.w #1,d2      (4)
            move.w d2,d0      (4)
lsqr4      movem.l  (sp)+,d1-d4 (44)
            rts              (16)

end

```

Listing Four

Listing Four

```

*****
*
*   Integer Square Root (32 to 16 bit).
*   (Newton-Raphson method).
*
*   Call with:
*       DO.L = Unsigned number.
*
*   Returns:
*       DO.L = SQRT(DO.L)
*
*   Notes:  Result fits in DO.W, but is valid in longword.
*           Takes from 338 cycles (1 shift, 1 division) to
*           1580 cycles (16 shifts, 4 divisions) (including rts).
*           Averages 854 cycles measured over first 65535 roots.
*           Averages 992 cycles measured over first 500000 roots.
*
*****
*
*   .globl lsqr
*
lsqr      movem.l  d1-d2,-(sp) (24)
            move.l d0,d1      (4)      ; Set up for guessing algorithm.
            beq.s  return     (10/8)   ; Don't process zero.
            moveq #1,d2      (4)

guess     cmp.l  d2,d1      (6)      ; Get a guess that is guaranteed to be
            bls.s  newton    (10/8)   ; too high, but not by much, by dividing the
            add.l  d2,d2      (8)      ; argument by two and multiplying a 1 by 2
            lsr.l  #1,d1      (10)     ; until the power of two passes the modified
            bra.s  guess     (10)     ; argument, then average these two numbers.

newton    add.l  d1,d2      (8)      ; Average the two guesses.
            lsr.l  #1,d2      (10)
            move.l d0,d1      (4)
            divu  d2,d1 (140)      ; Generate the next approximation(s)
            bvs.s  done (10/8)      ; via the Newton-Raphson method.
            cmp.w d1,d2      (4)      ; Handle out-of-range input (cheats!)
            bls.s  done (10/8)      ; Have we converged?
            swap  d1          (4)      ; No, kill the remainder so the
            clr.w d1          (4)      ; next average comes out right.
            swap  d1          (4)
            bra.s  newton    (10)

done      clr.w  d0          (4)      ; Return a word answer in longword.
            swap  d0          (4)
            move.w d2,d0      (4)

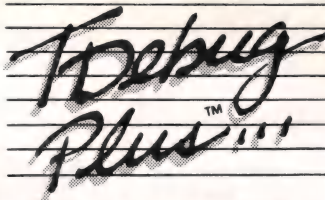
return    movem.l  (sp)+,d1-d2 (28)
            rts              (16)

end

```

End Listings

TURBO PROGRAMMERS-



... CUTS DEBUGGING FRUSTRATION.

TDEBUG-PLUS is a **new**, interactive symbolic debugger that integrates with Turbo Pascal to let you:

- **Examine and change variables** at runtime using symbolic names—including records, pointers, arrays, and local variables;
- **Trace and set breakpoints** using procedure names or source statements;
- **View source code** while debugging;
- **Use Turbo Pascal editor and DOS DEBUG commands.**

TDEBUG-PLUS also includes a special MAP file generation mode fully compatible with external debuggers such as Periscope, Atron, Symdeb, and others—even on programs written with Turbo EXTENDER.

An expanded, supported version of the acclaimed public domain program TDEBUG, the TDEBUG-PLUS package includes one DSDD disk, complete source code, a reference card, and an 80-page printed manual. 256K of memory required. Simplify debugging! \$60 COMPLETE.

TURBO EXTENDER™

Turbo EXTENDER provides you the following powerful tools to break the 64K barrier:

- **Large Code Model** allows programs to use all 640K without overlays or chaining, while allowing you to convert existing programs with minimal effort; makes EXE files;
- **Make Facility** offers separate compilation eliminating the need for you to recompile unchanged modules;
- **Large Data Arrays** automatically manages data arrays up to 30 megabytes as well as any arrays in expanded memory (EMS);
- **Additional Turbo EXTENDER tools** include Overlay Analyst, Disk Cache, Pascal Encryptor, Shell File Generator, and File Browser.

The Turbo EXTENDER package includes two DSDD disks, complete source code, and a 150-page printed manual. Order now! \$85 COMPLETE.

TURBOPOWER UTILITIES™

"If you own Turbo Pascal, you should own TurboPower Programmers Utilities, that's all there is to it." Bruce Webster, **BYTE Magazine**

TurboPower Utilities offers nine powerful programs: Program Structure Analyzer, Execution Timer, Execution Profiler, Pretty Printer, Command Repeater, Pattern Replacer, Difference Finder, File Finder, and Super Directory.

The TurboPower Utilities package includes three DSDD disks, reference card, and manual. \$95 with source code; \$55 executable only.

LIMITED TIME OFFER! BUY TWO OR MORE TURBOPOWER PRODUCTS AND SAVE 15%!

Satisfaction guaranteed or your money back within 30 days.

MC/VISA CALL TOLL-FREE 7 days a week
(US) 800-538-8157 x830
(CA) 800-672-3470 x830 For PD, COD,
Dealers, Info, Brochures—call or write:



478 W. Hamilton #196
Campbell, CA 95008
(408) 378-3672
M-F 9AM-5PM PST

The above TurboPower products require Turbo Pascal 3.0 (standard, 8087, or BCD) and PC DOS 2.X or 3.X, and run on the IBM PC/XT/AT and compatibles.

C CHEST

Listing One (Text begins on page 20.)

Listing 1 -- tree.h

```
1 typedef int      *TREE;                /* Dummy typedef for a tree. */
2
3 int      delete  ( TREE**, LEAF*, int(*)() );
4 LEAF     *insert ( TREE**, LEAF*, int(*)() );
5 LEAF     *find   ( TREE*,  LEAF*, int(*)() );
6
7 void     tprint  ( TREE*, int(*)(), FILE* );
8
9 LEAF     *talloc ( int );
10 void     tfree   ( LEAF* );
11 void     freeall ( TREE** );
```

End Listing One

Listing Two

Listing 2 -- test.c

```
1 #include <stdio.h>
2
3 typedef struct { int key; } LEAF;
4
5 #include "tree.h" /* LEAF must be defined before tree.h is #included */
6
7 /* ----- */
8
9 prnt( stream, p )
10 FILE *stream;
11 LEAF *p;
12 {
13     /* Print routine needed by tprint(). Should always print
14      * the same number of characters. When p == 0 it should
15      * print blanks.
16      */
17
18     fprintf( stream, p ? "%2d" : " ", p->key );
19 }
20
21 /* ----- */
22
23 icmp( n1, n2 )                /* Comparison routine for */
24 LEAF *n1, *n2;                /* insert() to use. */
25 {
26     return( n1->key - n2->key );
27 }
28
29 dcmp( key, n2 )                /* Comparison routine for */
30 LEAF *n2;                      /* delete() and find() to use. */
31 {
32     return( key - n2->key );
33 }
34
35 /* ----- */
36
37 docmd( cmd, n )
38 {
39     static TREE *root = NULL;
40     LEAF *p, *p2;
41
42     switch( cmd )
43     {
44     case 'd':
45         if( !delete(&root, (LEAF *) n, dcmp) )
46             fprintf(stderr, "Node NOT in tree\n");
47         break;
48
49     case 'f':
50         if( p = find(root, (LEAF *) n, dcmp) )
51             fprintf(stderr, "Node %d found\n", p->key );
52         else
53             fprintf(stderr, "Node NOT found\n" );
54         break;
55
56     case 'i':
57         if( !(p = talloc(sizeof(LEAF))) )
58             fprintf(stderr, "Out of memory.\n" );
59         else
60         {
61             p->key = n;
62             if( p2 = insert(&root, p, icmp) )
63                 {
```

Circle no. 207 on reader service card.


```

64         fprintf(stderr, "%d already in tree\n", p2->key);
65         tfree( p );
66     }
67     }
68     break;
69
70     case 'a':
71         freeall( &root );
72         break;
73
74     case 'q':
75         exit(0);
76     }
77
78     tprint( root, prnt, stdout );
79     printf("\n");
80 }
81
82 /* ----- */
83
84 main(argc, argv)
85 char **argv;
86 {
87     /* Assemble a tree, first get commands from the command line
88      * until these are exhausted, then get commands from the
89      * keyboard.
90      */
91
92     char buf[128];
93
94     for( ++argv; --argc > 0; ++argv )
95         docmd( **argv, atoi(*argv + 1) );
96
97     printf("commands are: iN -insert node N into tree\n");
98     printf("                dN -delete node N\n");
99     printf("                fN -find  node N\n");
100    printf("                a  -delete the entire tree\n");
101    printf("                q  -quit\n");
102
103    for(; gets(buf); printf("i/d/f/a/q: ") )
104        docmd( *buf, atoi(buf+1) );
105 }

```

End Listing Two

Listing Three

Listing 3 -- avl.h

```

1 typedef struct _leaf
2 {
3     struct _leaf *left ;
4     struct _leaf *right ;
5     unsigned size : 14 ;
6     unsigned bal : 2 ;
7 }
8 HEADER;
9
10     /* Possible values of bal field. Can be */
11     /* any three consecutive numbers but */
12     /* L < B < R must hold. */
13 #define L 0 /* Left subtree is larger */
14 #define B 1 /* Balanced subtree */
15 #define R 2 /* Right subtree is larger */
16
17 int delete ( HEADER**, HEADER*, int(*)() );
18 HEADER *insert ( HEADER**, HEADER*, int(*)() );
19 HEADER *find ( HEADER*, HEADER*, int(*)() );
20 void tprint ( HEADER*, int(*)(), FILE* );
21 HEADER *talloc ( int );
22 void tfree ( HEADER* );

```

End Listing Three

Listing Four

Listing 4 -- avlprnt.c

```

1 #include <stdio.h>
2 #include "avl.h"
3
4 /* -----
5  * These IBM graphics (box drawing) characters are used only if the
6  * output stream is stdout and isatty() is true (it will be false if
7  * stdout is redirected).
8  *
9  * GAMMA:  +---      ELL:  |      T RIGHT  +---
10 * \332    |          \300  +---      \303    |
11 *

```

(continued on next page)

HOW WOULD IT FEEL TO HAVE THE POWER OF TWO VAX[®] 11/780s ON YOUR DESK?

*With the Consulair[™]
Professional Development
Workstation, Now You Can!*

The programmers' dream system takes an ordinary Macintosh[™] and adds the Levco Prodigy 4, a 32 bit 68020 with a 68881 floating point coprocessor accelerated to 16 MHz, 4 megabytes of high speed memory, and SCSI interface. You get MacUser Magazine's Best Development Language of the Year, Consulair Mac C/ Mac C Toolkit with Direct Access[™] compiler support for the 68020 and 68881. Together, the system runs almost 1/2 million Whetstones per second and the Sieve benchmark in 0.68 seconds. As a special introductory offer you can upgrade your own Macintosh or Mac Plus for **\$6495**, including the complete software and hardware development system. An internal SCSI 20 Mbyte hard disk is available for an additional **\$995**.

If you want to move in smaller steps, we have a complete range of development systems to suit your needs. Leap into your future today with the Consulair Professional Development Workstation.

Call our Order and Information Hotline Today at **415-851-3272**.

Consulair

140 Campo Drive

Portola Valley, CA 94025

Circle no. 94 on reader service card.

OUTSIDE OKLAHOMA: NO SALES TAX

DYNAMIC RAM			
1M	1000Kx1	100 ns	\$70.00
256K	64Kx4	150 ns	4.00
256K	256Kx1	100 ns	5.50
256K	256Kx1	120 ns	3.40
256K	256Kx1	150 ns	2.95
128K	128Kx1	150 ns	4.37
64K	64Kx1	150 ns	1.40
EPROM			
27512	64Kx8	250 ns	\$23.00
27C256	32Kx8	250 ns	7.25
27256	32Kx8	250 ns	5.35
27128	16Kx8	250 ns	3.65
27C64	8Kx8	200 ns	4.55
2764	8Kx8	250 ns	3.40
STATIC RAM			
43256L-12	32Kx8	120 ns	\$35.00
6264LP-15	8Kx8	150 ns	3.10

OPEN 6 1/2 DAYS: WE CAN SHIP VIA FED-EX ON SAT.

Sunday & Holidays: Shipment or Delivery, via U.S. Express Mail \$11/2 lbs.

SAT DELIVERY INCLUDED ON FED-EX ORDERS RECEIVED BY: 24,000 S. Peoria Ave., BEGGS, OK. 74421 (918) 267-4961

Please call for current prices because prices are subject to change. Shipping & insurance extra. Cash discount prices shown. Orders received by 5 PM CST can usually be delivered to you by the next morning, via Federal Express Standard Air @ \$6.00, or Priority One @ \$13.00!

Circle no. 105 on reader service card.

Incredible SwyftWare!™

It is not possible in an ad to really explain this Apple II (e or c) product which outperforms all other software in speed and simplicity. It does no-frills word processing, information retrieval, telecommunications and many other tasks. Reviewers call it "foolproof and frustration free" and "blindingly fast."

It appeals to beginners and experts. Jef Raskin, who invented the Macintosh and who later created SwyftWare, has written up the hows and whys of this product. Please write for a copy. No charge. Or order SwyftWare and see for yourself. At \$89.95 and with a 30-day refund policy, there's no risk.

US 800/982-5600 CA 800/562-7400

Information Appliance

1014 Hamilton Ct., Menlo Park, CA 94025

New Release

CP/M ↔ **ISIS**
for
PDS & MDS

ICX v.4 eXchanger now supports BOTH 8" MDS and 5-1/4" iPDS formats. Manipulation of ISIS-II files using your CP/M system was never easier.

ISE v.6 Emulator gives the CP/M-80 user access to all the ISIS-II languages and utilities.

Complete source (C and MAC asm) included with all packages

ICXMDS\$89
ICXPDS\$89
ISE\$89
ICX Toolkit (all 3)\$250



Copyrights CP/M Digital Research, Inc.
ISIS-II and iPDS Intel Corp.

Western Wares

303-327-4898 • Box C • Norwood, CO 81423

Circle no. 269 on reader service card.

C CHEST

Listing Four (Listing continued, text begins on page 20.)

```

12 *
13 * T LEFT      ----+          T UP:  ----+          T DOWN  ----+
14 * \264        |          \331          \277          |
15 *
16 * VERT        |          (dash)
17 * \263        |          \304  -----
18 *
19 */
20
21 #define VERT    Cset[0]
22 #define GAMMA   Cset[1]
23 #define ELL     Cset[2]
24 #define T_LEFT  Cset[3]
25 #define T_UP    Cset[4]
26 #define T_DOWN  Cset[5]
27
28 #ifdef DEBUG
29 #   define PAD()    printf(" ");
30 #   define PBAL(r)  printf("(%c)", r->bal==B ? 'B': r->bal==L ? 'L': 'R');
31 #else
32 #   define PAD()
33 #   define PBAL(r)
34 #endif
35
36 /* ----- */
37
38 static char *Graph_chars[] = { "\263", "\332\304\304", "\300\304\304",
39                                "\304\304\264", "\304\304\331", "\304\304\277" };
40
41 static char *Norm_chars[] = { "|", "+--", "-+-", "--+", "--+", "--+" };
42
43 static int      (*Print)();      /* Node print function pointer */
44 static FILE     *Out;            /* Output stream */
45 static char     **Cset;         /* Current character set */
46 static char     Map[ 64/8 ];    /* Bitmap for 64 bits. If the
47                                /* tree is deeper than this,
48                                /* we're in trouble.
49
50 /* ----- */
51
52 #define testbit(c) ( Map[c >> 3] & (1 << (c & 0x07)) )
53
54 static setbit( c, val )
55 int c, val;
56 {
57     if( val )
58         Map[c >> 3] |= 1 << (c & 0x07);
59     else
60         Map[c >> 3] &= ~(1 << (c & 0x07));
61 }
62
63
64 /* ----- */
65
66 static trav( root, amleft )
67 HEADER *root;
68 int amleft;
69 {
70     /* Prints a binary tree graphically, with lines showing
71     * all the pointers. This is essentially the same routine
72     * we looked at last month. See that article for more
73     * info about how it works.
74     */
75
76     static int depth = -1;      /* Current depth in the tree */
77     static int i;
78
79     if( root )
80     {
81         ++depth;
82
83         if( root->right )
84             trav( root->right, 0 );
85         else
86             setbit( depth+1, 1 );
87
88         for( i = 1; i <= depth; i++ )
89         {
90             (*Print)( Out, 0 );
91             PAD();
92         }
93     }
94 }

```



```

93
94         if( i == depth )
95             fprintf(Out, " %s", amleft ? ELL : GAMMA );
96
97         else if( testbit(i) )
98             fprintf(Out, " %s ", VERT );
99         else
100             fprintf(Out, "   ");
101     }
102
103     (*Print)(Out, root + 1);
104     PBAL(root);
105
106     fprintf(Out, "%s\n",
107         (root->left) ? (root->right ? T_LEFT : T_DOWN)
108             : (root->right ? T_UP : "" )
109         );
110
111
112     setbit( depth, amleft ? 0 : 1);
113
114     if( root->left )
115         trav( root->left, 1 );
116     else
117         setbit( depth+1, 0 );
118
119     --depth;
120 }
121 }
122
123 /* ----- */
124
125 void tprint( root, print, stream )
126 HEADER *root;
127 int (*print)();
128 FILE *stream;
129 {
130     Out = stream;
131     Print = print;
132     Cset = Out == stdout && isatty(fileno(stdout)) ? Graph_chars
133         : Norm_chars ;
134     trav( root, 0 );
135 }

```

End Listing Four

Listing Five

Listing 5 -- avlfind.c

```

1 #include <stdio.h>
2 #include "avl.h"
3
4 HEADER *find( root, key, cmp )
5 HEADER *root;
6 HEADER *key;
7 int (*cmp)();
8 {
9     static int relation;
10
11     if( !root )
12         return NULL;
13
14     relation = (*cmp)( key, root + 1 );
15
16     return (relation == 0) ? root + 1 :
17         find( relation < 0 ? root->left : root->right, key, cmp );
18 }

```

End Listing Five

Listing Six

Listing 6 -- avlfree.c

```

1 #include <stdio.h>
2 #include "avl.h"
3
4 static void fa( root )
5 HEADER *root;
6 {
7     /* Delete the entire tree pointed to by root. Note that unlike
8      * tfree(), this routine is passed a pointer to a HEADER rather
9      * than to the memory just below the header.
10     */
11
12     if( root )
13     {

```

(continued on next page)

Work Smart with These Powerful C Utilities

Get more value from your C system. Boost program quality and slash development time with these professional utilities for leading C-compiler systems.

C Utility Library ~~\$185~~ \$155 Over 300 C subroutines

C and assembler source code and demonstration programs for screen handling, color printing, graphics, DOS disk and file functions, memory management and peripherals control.

C-tree ~~\$395~~ \$329 B-Tree database system

Store, update and retrieve records easily. High-level multi-key ISAM routines and low-level B-Tree functions. Available for MS-DOS, CP/M-86, and CP/M-80. Easily transported. Adaptable for network and multiuser. Includes source.

PHACT ~~\$295~~ \$200 Data Base Record Manager

Includes high-level features found in larger database systems. Available for MS-DOS, CP/M-86 and CP/M-80.

Pre-C ~~\$395~~ \$329 LINT-like source code analyzer

Locates structural and usage errors. Cross-checks multiple files for bad parameter declarations and other interface errors.

Windows for C ~~\$195~~ \$165 Versatile window utility

Supports IBM PC compatible and some non-compatible environments.

PANEL ~~\$295~~ \$235 Screen generating utility

Create custom screens via simple, powerful editing commands. Select colors, sizes and types, edit fields. Includes direct input utility.

HALO ~~\$280~~ \$199 Ultimate C graphics

A comprehensive package of graphics subroutines for C. Supports multiple graphics cards.

PLINK-86 ~~\$395~~ \$315 Overlay linker

Includes linkage editor, overlay management, a library manager and memory mapping. Works with Microsoft and Intel object format.

To order or for information call:

TECWARE
1-800-TEC-WARE
(In NJ call 201-530-6307)



UNIX is a registered TM of Bell Laboratories; C is a TM of AT&T; PHACT is a TM of PHACT ASSOC. INC.; PLINK-86, THE PHOENIX, HALO TM, MATH CYBERNETICS, INC.; PC-INT TM, GIMPLE, software, PANEL TM, Roundhill Computer Systems, Ltd.; WINDOWS FOR C TM, Creative Software, C/P M TM, TM DBI

Listing Six (Listing continued, text begins on page 20.)

```

14         fa( root->left );
15         fa( root->right );
16         free( root );
17     }
18 }
19
20 /*-----*/
21
22 void     freeall( root )
23 HEADER  **root;
24 {
25         fa( *root );
26         *root = NULL;
27 }

```

End Listing Six

Listing Seven

Listing 7 -- avlins.c

```

1 #include <stdio.h>
2 #include "avl.h"
3
4 /*-----*/
5 * Externally accessible routines:
6 *
7 * HEADER      *insert( rootp, newnode, cmp )   Insert newnode in tree
8 * HEADER      *talloc( size )                 Allocate a tree node
9 * void        tfree( p )                       Free a tree node
10 *
11 /*-----*/
12 */
13
14 static int    (*cmp)();
15 static HEADER *Newnode;
16 static HEADER *Conflicting;
17
18 /*-----*/
19
20 HEADER *talloc( size )
21 {
22     HEADER *malloc();
23     HEADER *p;
24
25     if( p = malloc( size + sizeof(HEADER)) )
26     {
27         p->left = NULL;
28         p->right = NULL;
29         p->size = size;
30         p->bal = B;
31         p++;
32     }
33     return p;
34 }
35
36 /*-----*/
37
38 void     tfree( p )
39 HEADER  *p;
40 {
41     free( --p );
42 }
43
44 /*-----*/
45
46 static int    ins( pp )
47 HEADER  **pp;
48 {
49     HEADER      *p;
50     HEADER      *p1, *p2;
51     int          relation;
52     /* relation > 0 <==> p > Newnode
53      * relation < 0 <==> p < Newnode
54      * relation == 0 <==> p == Newnode
55      */
56
57     static int    h = 0;
58     /* Set by recursive calls to search to
59      * indicate that the tree has grown.
60      * It will magically change its value
61      * everytime ins() is called recursively.
62      */

```

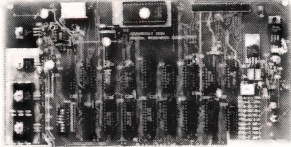
(continued on page 92)

DIGITAL RESEARCH COMPUTERS

(214) 225-2309

\$100 EPROM PROGRAMMER

OUR NEWEST DESIGN, FOR FAST EFFICIENT PROGRAMMING OF THE MOST POPULAR EPROM'S ON YOUR \$100 MACHINE. COMES WITH MENU DRIVEN SOFTWARE THAT RUNS UNDER CP/M 2.2 (8 INCH). PC BOARD SET CONSISTS OF (\$100) MAIN LOGIC BOARD REMOTE PROGRAMMING CARD AND SIX PERSONALITY MINI BOARDS FOR 2716, 2532, 2732, 2732A, 2764, AND 27128. SOLD AS BARE PC BOARD SET ONLY WITH FULL DOC. SOFTWARE FEATURES "FAST" PROGRAMMING ALGORITHM. FOR Z80 BASED SYSTEMS.



PC BOARD SET, FULL DOCUMENTATION, 8 IN. DISKETTE WITH SOFTWARE.

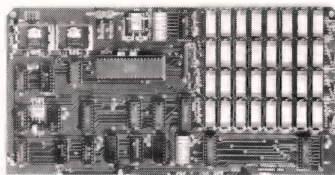
NEW! \$69.95

128K \$100 STATIC RAM/EPROM BOARD
JUST OUT! USES POPULAR 8K X 8 STATIC RAMS (6264) OR 2764 EPROMS. FOR 8 OR 16 BIT DATA TRANSFERS! IEEE 696 STANDARD. LOW POWER. KITS ARE FULLY SOCKETED. FULL DOC AND SCHEMATICS INCLUDED. 24 BIT ADDRESSING.

NEW! \$59.95 \$219.00 \$139.00
BARE PC BOARD 128K RAM KIT 128 EPROM KIT

256K S-100 SOLID STATE DISK SIMULATOR!
WE CALL THIS BOARD THE "LIGHT-SPEED-100" BECAUSE IT OFFERS AN ASTOUNDING INCREASE IN YOUR COMPUTER'S PERFORMANCE WHEN COMPARED TO A MECHANICAL FLOPPY DISK DRIVE.

PRICE CUT!



- FEATURES:**
- 256K on board, using + 5V 64K DRAMS.
 - Uses new Intel 8203-1 LSI Memory Controller.
 - Requires only 4 Dip Switch Selectable I/O Ports.
 - Runs on 8080 or Z80 S100 machines.
 - Up to 8 LS-100 boards can be run together for 2 Meg. of On Line Solid State Disk Storage.
 - Provisions for Battery back-up.
 - Software to mate the LS-100 to your CP/M* 2.2 DOS is supplied.
 - The LS-100 provides an increase in speed of up to 7 to 10 times on Disk Intensive Software.
 - Compare our price! You could pay up to 3 times as much for similar boards.

BLANK PCB
(WITH CP/M* 2.2
PATCHES AND INSTALL
PROGRAM ON DISKETTE)
\$49.95
(8203-1 INTEL \$29.95)

(ADD \$50 FOR A&T)

\$129.00

#LS-100

(FULL 256K KIT)

ZRT-80

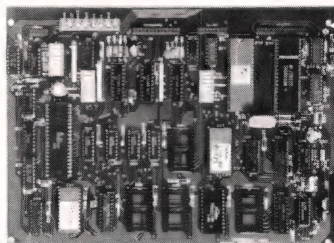
CRT TERMINAL BOARD!

A LOW COST Z-80 BASED SINGLE BOARD THAT ONLY NEEDS AN ASCII KEYBOARD, POWER SUPPLY, AND VIDEO MONITOR TO MAKE A COMPLETE CRT TERMINAL. USE AS A COMPUTER CONSOLE, OR WITH A MODEM FOR USE WITH ANY OF THE PHONE-LINE COMPUTER SERVICES.

FEATURES:

- Uses a Z80A and 6845 CRT Controller for powerful video capabilities.
- RS232 at 16 BAUD Rates from 75 to 19,200.
- 24 x 80 standard format (60 Hz).
- Optional formats from 24 x 80 (50 Hz) to 64 lines x 96 characters (60 Hz).
- Higher density formats require up to 3 additional 2K x 8 6116 RAMS.
- Uses N.S. INS 8250 BAUD Rate Gen. and USART combo IC.
- 3 Terminal Emulation Modes which are Dip Switch selectable. These include the LSI-ADM3A, the Heath H-19, and the Beehive.
- Composite or Split Video.
- Any polarity of video or sync.
- Inverse Video Capability.
- Small Size: 6.5 x 9 inches.
- Upper & lower case with descenders.
- 7 x 9 Character Matrix.
- Requires Par. ASCII keyboard.

FOR 8 IN. SOURCE DISK
(CP/M COMPATIBLE)
ADD \$10



\$89.95 A&T
#ZRT-80 ADD
(COMPLETE KIT, 2K VIDEO RAM) \$50

BLANK PCB WITH 2716
CHAR. ROM. 2732 MON. ROM
\$49.95

SOURCE DISKETTE - ADD \$10
SET OF 2 CRYSTALS - ADD \$7.50

64K \$100 STATIC RAM

\$99.00
KIT

LOW POWER!
150 NS ADD \$10

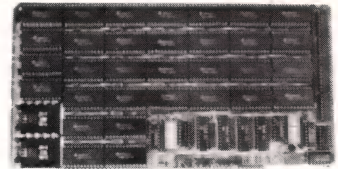
BLANK PC BOARD
WITH DOCUMENTATION
\$49.95

SUPPORT ICs + CAPS
\$17.50

FULL SOCKET SET
\$14.50

FULLY SUPPORTS THE
NEW IEEE 696 S100
STANDARD
(AS PROPOSED)

ASSEMBLED AND
TESTED ADD \$50



FEATURES: PRICE CUT!

- Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- Fully supports IEEE 696 24 BIT Extended Addressing.
- 64K draws only approximately 500 MA.
- 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- SUPPORTS PHANTOM (BOTH LOWER 32K AND ENTIRE BOARD).
- 2716 EPROMs may be installed in any of top 48K.
- Any of the top 8K (E000 H AND ABOVE) may be disabled to provide windows to eliminate any possible conflicts with your system monitor, disk controller, etc.
- Perfect for small systems since BOTH RAM and EPROM may co-exist on the same board.
- BOARD may be partially populated as 56K.

PANASONIC

Green Screen - Video Monitors

25 MHZ. TYPICAL BANDWIDTH!!!

Brand New In The Box! 9-Inch Screen

#K-904B1 (Chassis #Y08A) Open Frame Style

\$29.95

EA.

GROUP SPECIAL:

4 for \$99.00

WITH DATA & SCHEMATIC

(USA SHIPPING: \$3. PER UNIT. CANADA: \$7. PER UNIT)

COMPUTER MANUFACTURER'S EXCESS. STILL IN ORIGINAL PANASONIC BOXES. THE CRT TUBE ALONE WOULD COST MORE THAN OUR PRICE FOR THE COMPLETE UNIT. FOR SPLIT VIDEO (TTL INPUTS) OPERATION, NOT COMPOSITE VIDEO. OPERATES FROM 12VDC AT 1 AMP. VERTICAL INPUT IS 49 TO 61 HZ. HORIZONTAL INPUT: 15,750 HZ ± 500 HZ. RESOLUTION IS 800 LINES AT CENTER 650 LINES AT CORNERS.

THE NEW 65/9028 VT ANSI VIDEO TERMINAL BOARD!

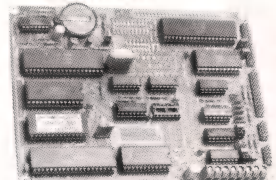
★ FROM LINGER ENTERPRISES ★

A second generation, low cost, high performance, mini sized, single board for making your own RS232 Video Terminal. Use as a computer console or with a MODEM for hook up to any of the telephone-line computer services.

FEATURES:

- Uses the new SMC 9028 Video Controller Chip coupled with a 6502A CPU.
- RS-232 at 16 Baud Rates from 50 to 19,200
- On board printer port!
- 24 X 80 format (50/60 Hz).
- For 15,750 Hz (Horiz.) monitors.
- 3 Terminal Modes: H-19, ADM3A, and ANSI X 3.64-1979
- Wide and thin-line graphics.
- White characters on black background or reversed.
- Character Attributes: De-Inten, Inverse, Underline and Blank.
- Low Power: 5VDC @ .7A, ± 12VDC @ 20MA.
- Mini size: 6.5 X 5 inches.
- Composite or split video.
- 5 X 8 Dot Matrix characters (U/L case) with descenders.
- Answer back capability.
- Battery backed up status memory.
- For ASCII parallel keyboard.

MICRO SIZE!



\$99.95

(Full Kit)

ADD \$40 FOR A&T

SOURCE DISKETTE:
PC/XT FORMAT
5 1/4 IN. \$15

OPTIONAL EPROM FOR
PC/XT STYLE SERIAL
KEYBOARD: \$15

Digital Research Computers

P.O. BOX 381450 • DUNCANVILLE, TX 75138 • (214) 225-2309

TERMS: Add \$3.00 postage. Orders under \$15 add 75¢ handling. No C.O.D. We accept Visa and MasterCard. Tex. Res. add 5-1/8% Tax. Foreign orders (except Canada) add 20% P & H. Orders over \$50 add 85¢ for insurance.

Listing Seven (Listing continued, text begins on page 20.)

```

61     if( !(p = *pp) )
62     {
63         p = Newnode ;      /* insert node in tree      */
64         h = 1;
65     }
66     else if( (relation = (* Cmp)( p+1, Newnode+1)) == 0 )
67     {
68         Conflicting = p + 1;
69     }
70     else if( relation > 0 )
71     {
72         ins( &p->left );
73
74         if( h )              /* left branch has grown */
75         {
76             switch( p->bal )
77             {
78                 case R: p->bal = B ; h = 0; break;
79                 case B: p->bal = L ;      break;
80
81                 case L:              /* rebalance */
82                     p1 = p->left;
83                     if( p1->bal == L )    /* Single LL */
84                     {
85                         p->left = p1->right;
86                         p1->right = p;
87                         p->bal = B;
88                         p = p1;
89                     }
90                     else              /* Double LR */
91                     {
92                         p2 = p1->right;
93                         p1->right = p2->left;
94                         p2->left = p1;

```

(continued on page 94)

UNIX Tools on DOS

MKS Toolkit

Building blocks for the UNIX professional working in a DOS environment

The MKS Toolkit consists of over 60 programs that perform tasks on machines like the IBM PC, XT, or AT with the ease that one would expect while working under UNIX. Designed especially for those developing software in a DOS environment, these utilities include:

- prof** — give a profile of the execution times of a command
- egrep** — find a string using full regular expression patterns
- awk** — data transformation & report generation language
- diff** — find differences between two files

cat	chmod	cmp	comm	cp	cut	date	dd	dev
df	du	ed	echo	file	find	head	help	lc
line	ls	mv	nm	od	paste	rm	sed	sh
size	split	strings	tail	time	touch	tr	uniq	wc

The programs come with a shell and complete glob facility (for UNIX-style command-line file name expansion) on 2 DSDD 5.25" floppies, load and run under DOS, and are not copy-protected. Phone support is available during business hours. Full documentation is included.

Price: \$99 from: Mortice Kern Systems Inc.,
43 Bridgeport Rd. E., Waterloo, Ontario N2J 2J4

For information or ordering call collect: **519-884-2251**

MasterCard & VISA orders accepted. OEM & dealer inquiries invited.
UNIX is a trademark of Bell Labs. MS-DOS is a trademark of Microsoft Corp.

STREAMLINE YOUR PROGRAMMING

Circle no. 249 on reader service card.

DEC & DG Software Tools

C68K C Cross Compiler for Motorola 680X0 development on DEC and DG CPUs. Executable code from C source. Complete with compiler, assembler, linker, librarian, and the advanced debugger lint-PLUS.

lint-PLUS Comprehensive C Debugger using high level techniques; finds many bugs before traditional debugging even begins. Global source analysis, runtime batch and interactive modes as well.

DG-C K & R standard; the only 16-bit C Compiler for all Data General operating systems.

F68K Fortran Cross Compiler for Motorola 680X0 development on DEC and DG CPUs. Executable code from Fortran source. Complete with compiler, assembler, linker, librarian, and the source debugger FORTRAN-lint.

FORTTRAN-lint A pre-compile source code analyzer that globally detects inconsistencies in common block declarations or in variable & argument lists between modules; unused variables & functions, variable type usage conflicts and many other problem elements.

Call or write today for details!

IPT CORPORATION
1096 E. Meadow Circle Palo Alto, CA 94303
(415) 494-7500

Circle no. 252 on reader service card.

Now You Know Why **BRIEF**™ is **BEST**

"BRIEF is simple to learn and use and extremely sophisticated."

PC Magazine, July 1986

The Program Editor with the **BEST** Features

Since its introduction, BRIEF has been sweeping programmers off their feet. Why? Because BRIEF offers the features **MOST ASKED FOR** by professional programmers. In fact, BRIEF has just about every feature you've ever seen or imagined, including the ability to configure windows, keyboard assignments, and commands to **YOUR** preference. One reviewer (David Irwin, DATA BASED ADVISOR) put it most aptly, "(BRIEF)... is quite simply the best code editor I have seen."

COMPILER SUPPORT

No matter what compiler you have, it will run inside BRIEF. If errors occur during compilation, the supplied macros place your cursor on the line with the first problem and display the compiler's message. After you make your corrections, you skip to the next error with one keystroke. BRIEF automatically moves your cursor to the right place, even if you've added or deleted lines.

BRIEF is preconfigured (using the built-in macro language) for the Microsoft Macro Assembler v 4.0, and the Microsoft, Computer Innovations, Lattice, and Wizard C compilers. If you use another product, you can modify the macros to support it.

Every Feature You Can Imagine

Compare these features with your editor (or any other for that matter).

- FAST
- Full UNDO (N Times)
- Edit Multiple Large Files
- Compiler-specific support, like auto indent, syntax check, compile within BRIEF, and template editing
- Exit to DOS inside BRIEF
- Uses all Available Memory
- Tutorial
- Repeat Keystroke Sequences
- 15 Minute Learning Time
- Windows (Tiled and Pop-up)
- Unlimited File Size – (even 2 Meg!)
- Reconfigurable Keyboard
- Context Sensitive Help
- Search for "regular expressions"
- Mnemonic Key Assignments
- Horizontal Scrolling
- Comprehensive Error Recovery
- A Complete Compiled Programmable and Readable Macro Language
- EGA and Large Display Support
- Adjustable line length up to 512

Program Editing **YOUR** Way

A typical program editor requires you to adjust your style of programming to its particular requirements – NOT SO WITH BRIEF. You can easily customize BRIEF to your way of doing things, making it a natural extension of your mind. For example, you can create ANY command and assign it to ANY key – even basic function keys such as cursor-control keys or the return key.

The Experts Agree

Reviewers at BYTE, INFO WORLD, DATA BASED ADVISOR, and DR. DOBB'S JOURNAL all came to the same conclusion – **BRIEF IS BEST!**

Further, of 20 top industry experts who were given BRIEF to test, 15 were so impressed they scrapped their existing editors!

NOT COPY PROTECTED

Solution Systems™

MONEY-BACK GUARANTEE

Try BRIEF (\$195) for 30 days – If not satisfied get a full refund.

TO ORDER CALL (800-821-2492)

SOLUTION SYSTEMS, 335-D WASHINGTON ST., NORWELL, MA 02061, 617-659-1571

BRIEF is a trademark of UnderWare

Choose your weapon ... ForthCard or C Board

Our single board computers
allow you to write high level
embedded applications.

HiTech Equipment
Corporation

9560 Black Mountain Road
San Diego, CA 92126
(619) 566-1892

Circle no. 196 on reader service card.

IBM / PC CROSS ASSEMBLERS

Assemblers now available include:

Chip	Chip	Chip
1802/1805	HD64180	8051
9900/9995	NSC800	6804
6500/01/02	F8, 3870	6805
6800/01/02	Z8	6809
6800/08/10	Z80	6811
8048/49/50/42	Z8000	8085
65C02/C102/C112	6803/08	6301

RELATIONAL MEMORY SYSTEMS, INC.
P. O. Box 6719
San Jose, California 95150

Tel: (408) 265-5411
CPM80, MPM, ISIS Versions Available

relms

Circle no. 120 on reader service card.

C Users' Group

Over 90 volumes of public
domain "C" software including:

- compilers
- editors
- text formatters
- communications packages
- many UNIX-like tools

Write or call for more details

The C Users' Group

Post Office Box 97
McPherson, KS 67460
(316) 241-1065

Circle no. 181 on reader service card.

C CHEST

Listing Seven (Listing continued, text begins on page 20.)

```

95         p->left  = p2->right;
96         p2->right = p;
97         p->bal   = (p2->bal == L) ? R : B ;
98         p1->bal  = (p2->bal == R) ? L : B ;
99         p        = p2;
100
101         }
102         p->bal = B;
103         h     = 0;
104     }
105 }
106 else
107 {
108     ins( &p->right );
109
110     if( h )                /* right branch has grown */
111     {
112         switch( p->bal )
113         {
114             case L: p->bal = B;   h = 0;   break;
115             case B: p->bal = R;   break;
116
117             case R:                /* rebalance: */
118                 pl = p->right;
119                 if( pl->bal == R ) /* Single RR */
120                 {
121                     p->right = pl->left;
122                     pl->left = p;
123                     p->bal   = B;
124                     p        = pl;
125                 }
126                 else                /* Double RL */
127                 {
128                     p2      = pl->left;
129                     pl->left = p2->right;
130                     p2->right = pl;
131                     p->right = p2->left;
132                     p2->left = p;
133                     p->bal   = (p2->bal == R) ? L : B ;
134                     pl->bal  = (p2->bal == L) ? R : B ;
135                     p        = p2;
136                 }
137                 p->bal = B;
138                 h     = 0;
139             }
140         }
141     }
142
143     *pp = p;
144 }
145
146 /*-----*/
147
148 HEADER *insert( rootp, newnode, cmp )
149 HEADER **rootp;
150 HEADER *newnode;
151 int     (*cmp) ();
152 {
153     /* Insert newnode into tree pointed to by *rootp. Cmp is passed
154      * two pointers to HEADER and should work like strcmp().
155      * Return NULL on success or a pointer to the conflicting node
156      * on error.
157      */
158
159     Cmp      = cmp;
160     Newnode  = newnode - 1;
161     Conflicting = NULL;
162
163     ins(rootp);
164
165     return Conflicting;
166 }

```

End Listing Seven

Listing Eight

Listing 8 -- avl1.c

```

1 #include <stdio.h>
2 #include "avl.h"

```

(continued on page 96)



MULTIPLE CHOICE

TECH PC

MULTIUSER SYSTEM



FEATURES: TECH PC TURBO QUAD BUSINESS SYSTEM Starting from \$5999

Tech PC/XT base unit in portable or desktop configuration with 640K, multiple serial ports, three Tech PC terminals, connecting cables, and networking software. Separate NEC V20 8088 Intel compatible 8 MHz CPU and up to 1 MB RAM for each terminal on the system.

Two fully functional serial ports per terminal.

Four users expandable to 32 users over dumb terminals or PC's with terminal emulation software. Capacity for unlimited number of local printers.

Full support for multitasking multiterminal use with print spooling for multiple printers, background monitoring of the system, dial up bulletin board support, password protection, and file/record locking supporting PC network protocol.

System supports all popular software such as Wordstar, dBaseIII, Lotus 123, Multimate, etc.

FEATURES: TECH PC TRIAD MULTIUSER Starting from \$2599

Tech PC/XT base unit with 640K, and two 360K disk drives.

Separate Intel 80188 Microprocessor running at 8 MHz and 512K for each terminal.

Three high resolution monitors, three Selectric style Hi-Tek keyboards, 50 feet of shielded cable to separate the three stations.

System expandable to 32 workstations.

System supports up to six printers.

Full support for multitasking multiterminal use with print spooling for multiple printers, background monitoring of the system, dial up bulletin board support, password protection, and file/record locking supporting PC network protocol.

System supports all popular software such as Wordstar, dBaseIII, Lotus 123, Multimate, etc.

FEATURES: TECH PC TWIN MULTIUSER Starting from \$1699

Tech PC/XT base unit with 640K, and two 360K disk drives.

Two high resolution monitors, two Selectric style Hi-Tek keyboards, 50 feet of shielded cable to separate the two stations.

System supports up to six printers.

Full software support with multi-level file security, electronic message facility to send and receive messages between users, password logon system, and system operator command level.

System supports all popular software such as Wordstar, dBaseIII, Lotus 123, Multimate, etc.



FEATURES: TECH PC QUAD BUSINESS SYSTEM Starting from \$4499

Tech Turbo PC/AT base unit in portable or desktop configuration with 512K, multiple serial ports, three Tech PC terminals, connecting cables, and networking software. Four users expandable to nine users over dumb terminals or PC's with terminal emulation software.

Capacity for up to 16 printers at remote sites with up to 6 local printers attached to the main unit.

Each user can access 512K of RAM with memory expansion boards.

Full support for multitasking multiterminal use with print spooling for central or terminal printing, background monitoring of the system, dial up bulletin board support, password protection, and file/record locking using PC network protocol.

System supports all popular software such as Wordstar, dBaseIII, Lotus 123, Multimate, etc.

THIRD PARTY MAINTENANCE AVAILABLE THROUGH MOMENTUM SERVICES CORP.

Users only circle no. 279 on reader services card. Dealers only circle no. 245 on reader service card.

TECH PC

TECH PERSONAL COMPUTERS
2131 South Hathaway, Santa Ana, California 92705

TELEX: 272006 Answer Back-TECH

714/754-1170

FAX: 714/556-8325

PLEASE ALLOW ONE WEEK FOR SHIPPING

VISA, MASTERCARD

Listing Eight (Listing continued, text begins on page 20.)

```

3
4 /*-----
5 *      Local static subroutines:
6 */
7
8 extern int      balance_l ( HEADER** );
9 extern int      balance_r ( HEADER** );
10 extern int      descend  ( HEADER**, HEADER** );
11 extern int      del      ( HEADER** );
12
13 /*-----*/
14
15 static HEADER *Key;
16 static int      (*Cmp) ();
17 static int      Notfound;
18
19 /*-----*/
20
21 static int      balance_l ( pp )
22 HEADER **pp;
23 {
24     /* This routine is called when the left branch of the current
25      * subtree (pointed to by p) has shrunk. It adjusts the balance
26      * factors and rebalances if necessary, modifying *pp to point
27      * at the new root (after the rebalance). Returns 1 if the
28      * tree got smaller as a result of the delete or the rebalance
29      * operation, else returns 0.
30      */
31
32     register HEADER *p, *p1, *p2;
33     int              b1, b2;
34     int              got_smaller = 1;
35
36     p = *pp;
37
38     switch( p->bal )
39     {
40     case L: p->bal = B; break;
41     case B: p->bal = R; got_smaller = 0; break;
42     case R: break; /* Rebalance */
43
44         p1 = p->right;
45         b1 = p1->bal;
46
47         if( b1 >= B ) /* Single RR */
48         {
49             p->right = p1->left;
50             p1->left = p;
51
52             if( b1 != B )
53                 p->bal = p1->bal = B;
54             else
55             {
56                 p->bal = R;
57                 p1->bal = L;
58                 got_smaller = 0;
59             }
60             p = p1;
61         }
62         else
63         {
64             p2 = p1->left; /* Double RL */
65             b2 = p2->bal;
66             p1->left = p2->right;
67             p2->right = p1;
68             p->right = p2->left;
69             p2->left = p;
70             p->bal = (b2 == R) ? L : B;
71             p1->bal = (b2 == L) ? R : B;
72             p = p2;
73             p2->bal = B;
74         }
75
76         *pp = p;
77         return got_smaller;
78     }
79
80 /*-----*/
81
82 static int      balance_r ( pp )

```

(continued on page 98)

C & PASCAL PROGRAMMERS

Blaise Computing provides a broad range of programming tools for Pascal and C programmers, with libraries designed for serious software development. You get carefully crafted code that can be easily modified to grow with your changing needs. Our packages are shipped complete with comprehensive manuals, sample programs and source code.

C TOOLS PLUS

\$175.00

NEW! Full spectrum of general-purpose utility functions; windows that can be stacked, removed, and accept user input; interrupt service routines for resident applications; screen handling including EGA 43-line text mode support and direct screen access; string functions; and DOS file handling.

PASCAL TOOLS/TOOLS 2

\$175.00

Expanded string and screen handling; graphics routines; easy creation of program interfaces; memory management; general program control; and DOS file support.

VIEW MANAGER

\$275.00

Complete screen management; paint data entry screens; screens can be managed by your application program; block mode data entry or field-by-field control. Specify C or IBM/MS-Pascal.

ASYNCH MANAGER

\$175.00

Full featured asynchronous communications library providing interrupt driven support for the COM ports; I/O buffers up to 64K; XON/XOFF protocol; baud rates up to 9600; modem control and XMODEM file transfer. Specify C or IBM/MS-Pascal.

Turbo POWER TOOLS PLUS

\$99.95

NEW! Expanded string support; extended screen and window management including EGA support; pop-up menus; memory management; execute any program from within Turbo Pascal; interrupt service routine support allowing you to write memory resident programs; schedulable intervention code.

Turbo ASYNCH PLUS

\$99.95

Complete asynchronous communications library providing interrupt driven support for the COM ports; I/O buffers up to 64K; XON/XOFF protocol; and baud rates up to 9600.

RUNOFF

\$49.95

NEW! Text formatter written especially for programmers; flexible printer control; user-defined variables; index generation; and general macro facility. Crafted in Turbo Pascal.

EXEC

\$95.00

Program chaining executive. Chain one program from another even if the programs are in different languages. Shared data areas can be specified.

ORDER TOLL-FREE 800-227-8087!



BLAISE COMPUTING INC.

2560 Ninth Street, Suite 316 Berkeley, CA 94710 (415) 540-5441

Circle no. 217 on reader service card.

Dr. Dobb's Journal, August 1986



NEW! FROM BLAISE COMPUTING

Today's programmers need more than yesterday's tools. Requirements such as removable windows and "sidekickable" pop-up utilities are changing the face of program design. You need to filter interrupts so that other resident programs still work. You need the ability to switch between multiple display pages and monitors. Today's technical demands are almost endless, but C TOOLS PLUS gives you what you need.



SOLID LIBRARY SUPPORT

Blaise Computing offers you solid library support that can meet all your demands and more. C TOOLS PLUS embodies the full spectrum of general-purpose utility functions that are critical to today's applications.

Here's just part of the PLUS in C TOOLS PLUS:

- ◆ **C TOOLS and C TOOLS 2** compatibility — two packages that receive rave reviews for quality, organization, usability and documentation.
- ◆ **FULL SOURCE CODE**

C Tools PlusTM

For The Programmer Whose Alphabet Begins & Ends With "C"



- ◆ **WINDOWS** that are stackable, removable, that support word wrap and that can accept user input.
- ◆ **INTERRUPT SERVICE ROUTINE** support for truly flexible, robust and polite resident applications.
- ◆ **MULTIPLE** monitor and display support, including EGA 43-line mode.
- ◆ **FAST DIRECT VIDEO ACCESS** for efficiency that will not constrain good program design.
- ◆ **DOCUMENTATION, TECHNICAL SUPPORT** and attention to detail that have distinguished Blaise Computing products over the years.

C TOOLS PLUS supports the Microsoft (and IBM) 3.00 and Lattice 3.00 C compilers and is just \$175.00.

Also Available Are:
C VIEW MANAGER — A kit for building data entry screens and menus. Begin by designing on-screen what the operator will see; call upon our library functions from your program to display the screens and retrieve the data. Just \$275, including all library source code.

C ASYNCH MANAGER — provides the crucial core of hardware interrupt support needed to build applications that communicate. It

also includes the "XMODEM" file-transfer protocol and support for Hayes-compatible modems. All source code is included for \$175. **C TOOLS & C TOOLS 2** — an indispensable combination still available at a low price of \$175, including all source code. See review in PC Tech Journal, 6/85.

BLAISE COMPUTING INC.

2560 Ninth Street, Suite 316 Berkeley, CA 94710 (415) 540-5441

ORDER TOLL-FREE 800-227-8087!

YES, send me the PLUS I need! Enclosed is \$_____ for C TOOLS PLUS. (CA residents add 6½% Sales Tax. All domestic orders add \$10.00 for Federal Express shipping.)
Name: _____ Phone: (____) _____
Shipping Address: _____ State: _____ Zip: _____
City: _____ Exp. Date: _____
VISA or MC #: _____

Listing Eight (Listing continued, text begins on page 20.)

```

83 HEADER  **pp;
84 {
85     /*      Same as balance_l but is called when a right subtree
86     *      has been made smaller.
87     */
88
89     register HEADER *p, *p1, *p2;
90     int      b1, b2;
91     int      got_smaller = 1;
92
93     p = *pp;
94
95     switch( p->bal )
96     {
97     case R: p->bal = B;          break;
98     case B: p->bal = L;  got_smaller = 0;  break;
99     case L:                                     /* rebalance */
100         p1 = p->left;
101         b1 = p1->bal;
102
103         if( b1 <= B )          /* Single LL */
104         {
105             p->left  = p1->right;
106             p1->right = p;
107
108             if( b1 != B )
109                 p->bal = p1->bal = B;
110             else
111             {
112                 p->bal  = L;
113                 p1->bal  = R;
114                 got_smaller = 0;
115             }
116             p = p1;
117         }
118         else

```

(continued on page 100)

MacTutor

The Macintosh Programming Journal



Simply the finest monthly technical Journal available for the Macintosh. Contains no fluff, only programs in C, Asm, Pascal, Basic & Forth. US 3rd class: \$30; US 1st class, or Canada: \$45. All Overseas: \$60. (714) 630-3730.

Subscribe
Today!

MacTutor
P.O. Box 400
Placentia, CA. 92670

FORTRAN PROGRAMMERS

Looking for the right PC FORTRAN LANGUAGE SYSTEM? If you're serious about your FORTRAN programming then you should be using F77L- LAHEY FORTRAN.

Editor's Choice - PC Magazine

- Full FORTRAN 77 Standard (F77L is not a subset)
- Popular Extensions for easy porting of minicomputer and mainframe applications
- COMPLEX*16, LOGICAL*1 and INTEGER*2
- Recursion - allocates local variables on the stack
- IEEE - Standard Floating Point Arithmetic
- IMPLICIT NONE
- Long variable names - 31 characters
- Fast Compile - Increase your productivity
- Source on Line Debugger (Advanced features without recompiling)
- Arrays and Common Blocks greater than 64K
- Clear and Precise English Diagnostics
- Compatibility with Popular 3rd Party Software (i.e. Lattice C)
- Easy to use manual
- Technical Support from LCS

F77L - THE PROGRAMMER'S FORTRAN \$477.00 U.S.

System Requirements: MS-DOS or PC-DOS, 256K, math coprocessor (8087/80287)

FOR MORE INFORMATION: (702) 831-2500



Lahey Computer Systems Inc.

P.O. Box 6091 Incline Village, NV 89450/USA

International Dealers:

England:	Grey Matter Ltd.,	Tel: (0364) 53499
Denmark:	Ravenholm Computing,	Tel: (02) 887249
Australia:	Computer Transitions	Tel: (03) 537-2786
Japan:	Microsoft, Inc.,	Tel: (03) 813-8222

SERVING THE FORTRAN COMMUNITY SINCE 1967

The Full-Service Source For Programming Software

At Lifeboat, we're committed to more than selling you the best software. We provide tech support for every program we sell. Our expert staff is ready to take your calls on our Tech Support Hotline and help you quickly solve any technical problems you may encounter. Call Lifeboat for the complete solution to your programming needs.

LANGUAGES

Lattice C 3.0 The best selling C compiler has been upgraded to give you more functions and features. Lattice C 3.0 contains 200 new library functions, better code generation, support for new data types (void, enum, unsigned char, unsigned long), support for the 80186/80286 instruction set and the ability to generate in-line 8087/80287 instructions. Lattice C is the C compiler for professional developers.

RUN/C—The C Interpreter Learn C the natural way with RUN/C. The user interface is similar to BASIC with easy, familiar commands. The new 2.0 version of RUN/C comes with a full-screen editor and other enhancements.

RUN/C Professional New RUN/C Professional "... is the overall best choice of a C interpreter." *PC Tech Journal* (5/86). All RUN/C's capabilities plus powerful features for program development. Load your favorite object libraries with RUN/C Professional. Contains a full-screen editor and source code debugging facilities.

Pro Pascal A truly standard Pascal. Produces fast, tight code with plenty of compile time options.

BetterBASIC New Version Now you can program in BASIC and use the full memory of your PC, create structured programs using functions and procedures, make your own library modules and more. Now compatible with Microsoft BASIC.

LANGUAGE UTILITIES

Plink86 Plus An overlay linkage editor for linking 8086/8088 object modules. Contains new features for memory caching, library allocation, file merging and overlay reloading.

Pfix86 Plus A symbolic and source level advanced debugger for programming professionals.

Periscope I & II The symbolic debugger that can debug anywhere within a program at anytime with a simple press of a button. Periscope's breakout button will interrupt infinite loops, gain freedom from a locked up keyboard or help you find out where that slow running program is spending all its time. Periscope I additionally features a protected memory board which shields the debugger from runaway programs.

BASTOC

A BASIC to C translator for the BASIC programmer who wants to upgrade to C.

Circle no. 118 on reader service card.

EDITORS

Brief This programmer's editor has several outstanding features. Brief's macro language compiler allows the programmer to create new commands and assign them to any key. Brief's unique "undo" facility will even let you recover from accidentally entered commands. Create as many windows as will fit on a screen, containing the same or different portions of one or several files. Compile programs without ever having to leave Brief.

FirTime Speed-up program development with an easy-to-use C and Pascal syntax checking editor. Detects syntax errors, undefined variables and mismatched type assignments. Use function key menus to generate statements or enter your code directly. On-line help available.

VEDIT Plus A full-screen text editor for program development and word processing. It contains powerful features including use of macros, on-line help facility, paragraph formatting, and file comparison.

EMACS Customizable editor including windowing, multi-tasking and special modes for C and Pascal.

FUNCTIONS

The Greenleaf Comm Library

A library of over 120 communication routines. Contains functions to create interrupt driven routines or perform direct I/O to multiple Comm Ports. Its strengths are in asynchronous communications, interrupt mode, modem control, XMODEM, XON/XOFF and flow control.

PforCe New A library of 400 plus functions for interrupt driven communications, background tasks, string/table parsing, data base manipulation, DOS and BIOS functions and more. Superbly written and documented software with complete source code included.

The Greenleaf Functions A mature library of over 200 functions. Version 3.0 offers all new indexed documentation, with an abundance of examples. Source code included.

Essential C Utility Library Over 300 functions, with special attention given to screen handling, windows and business graphics. Source code is included.

BASIC.C Speed-up development time with this BASIC to C library, which acts as a bridge to C providing many of the capabilities of the BASIC language.

GRAPHICS, WINDOWING and SCREEN DESIGN

GSS*CGI GRAPHICS New The GSS Computer Graphics Interface is designed for creating high performance graphics-based applications. GSS*CGI speeds up application development and provides compatibility with a wide range of peripherals. It's the only CGI implementation that provides true device-independence for both raster and vector graphics. Products in the GSS GRAPHICS line include: the GSS*CGI Graphics Development Toolkit, the GSS Kernel System; the GSS Plotting System; and the GSS*CGI Metafile Interpreter.

Essential Graphics New A brand new graphics library for C programmers with the emphasis placed on ease-of-use and portability. No royalties.

WINDOWS FOR DATA New Provides C programmers with complete control over screen display and data entry, within an easy-to-use windowing system. Features include one-step data entry, multiple data types, Lotus-style menu design, full-featured field editing, pop-up entry windows, and much more.

Panel A powerful tool for interactive screen design.

For more information on these and other products in our complete line call:

1-800-847-7078

NY: 914-332-1875

Special Introductory Offer! For a limited time only you can get the new Software Programming Tools Guide from Lifeboat **FREE** by returning this coupon. To receive your copy of this regular \$9.95 value, you must supply all of the information below. Sophisticated programmers won't want to be without this complete guide to product features and selection.

Name _____ Title _____
Company _____ Business phone _____
Address _____ Zip _____

The Full-Service Source for Programming Software

LIFEBOAT

The names of products listed are generally the trademarks of the sources of the products.

© 1986 Lifeboat Associates

INTERNATIONAL
SALES OFFICES

Australia
Fagan Microprocessor Associates
Phone (61) 3589-9889
Canada
Scintel Systems
Phone (416) 449-8752
England
Grey Matter Ltd.
Phone (44) 364-53489
Italy
Lifeboat Associates, S.p.A.
Phone (02) 656-841
Japan
Lifeboat Japan
Phone (03) 293-4711
Spain
Micronet, S.A.
Phone (34) 1-457-5056
The Netherlands
GIGA Computer Products
Phone (31) 30-771846
West Germany
MEMA Computer GmbH
Phone (49) (069) 34 72 26
Omnia
Phone (49) (076) 23 61820

Lifeboat
55 South Broadway
Tarrytown, NY 10591

Instant-C:TM The Fastest Interpreter for C

**Runs your programs 50
to 500 times faster than
any other C language
interpreter.**

Any C interpreter can save you compile and link time when developing your programs. But only **Instant-C** saves your time by running your program at compiled-code speed.

Fastest Development. A program that runs in one second when compiled with an optimizing compiler runs in two or three seconds with **Instant-C**. Other interpreters will run the same program in two minutes. Or even ten minutes. Don't trade slow compiling and linking for slow testing and debugging. *Only Instant-C will let you edit, test, and debug at the fastest possible speeds.*

Fastest Testing. **Instant-C** immediately executes any C expression, statement, or function call, and display the results. Learn C, or test your programs faster than ever before.

Fastest Debugging. **Instant-C** gives you the best source-level debugger for C. Single-step by source statement, or set any number of conditional breakpoints throughout your program. Errors always show the source statements involved. Once you find the problem, test the correction in seconds.

Fastest Programming. **Instant-C** can directly generate executable files, supports full K & R standard C, comes with complete library source, and works under PC-DOS, MS-DOS, or CP/M-86. *Instant-C gives you working, well-tested programs faster than any other programming tool.* Satisfaction guaranteed, or your money back in first 31 days. **Instant-C** is \$495.

Rational
Systems, Inc.

P.O. Box 480
Natick, MA 01760
(617) 653-6194

C CHEST

Listing Eight (Listing continued, text begins on page 20.)

```

119      {
120          p2      = p1->right;          /* Double RL */
121          b2      = p2->bal;
122          p1->right = p2->left;
123          p2->left  = p1;
124          p->left  = p2->right;
125          p2->right = p;
126          p->bal   = (b2 == L) ? R : B ;
127          p1->bal  = (b2 == R) ? L : B ;
128          p       = p2;
129          p2->bal  = B;
130      }
131  }
132
133      *pp = p;
134      return got_smaller ;
135  }
136
137  /* ----- */
138
139  static int descend( rootp, dpp )
140  HEADER **rootp ;          /* Address of root of current node */
141  HEADER **dpp ;           /* Address of node to be deleted */
142  {
143      /* Does the actual delete when the root node has both left and
144      * right descendants. Descends to the rightmost node of the
145      * left subtree and then copies the contents of that node
146      * to the node-to-be-deleted (*dpp). Then the node-to-be-deleted
147      * is modified to point at the former rightmost node.
148      */
149
150      if( (*rootp)->right )
151          return( descend( &(*rootp)->right, dpp )
152                  ? balance_r(rootp) : 0 ;
153
154      else
155      {
156          memcpy( *dpp + 1, *rootp + 1, (*rootp)->size );
157
158          *dpp      = (*rootp);
159          *rootp     = (*rootp)->left;
160          return 1;
161      }
162  }
163  /* ----- */
164
165  static int del( rootp )
166  HEADER **rootp;
167  {
168      /* Delete Key from tree pointed to by *rootp. Return 1 if the size
169      * of the tree has been reduced, 0 otherwise.
170      */
171
172      HEADER *dp;          /* Pointer to node to delete */
173      int got_smaller = 0;  /* set TRUE if tree shrinks */
174      static int relation;
175
176      if( !*rootp )
177          Notfound = 1;
178      else
179      {
180          relation = (*Cmp)(Key, *rootp + 1);
181
182          if( relation < 0 )          /* Go left */
183          {
184              if( del( &(*rootp)->left ) )
185                  got_smaller = balance_l( rootp );
186          }
187          else if( relation > 0 )      /* Go right */
188          {
189              if( del( &(*rootp)->right ) )
190                  got_smaller = balance_r( rootp );
191          }
192          else
193              /* Delete current */
194          {
195              dp = *rootp ;
196
197              if( dp->right == NULL )
198              {
199                  *rootp      = dp->left;
200                  got_smaller = 1;

```



```

200     }
201     else if( dp->left == NULL )
202     {
203         *rootp      = dp->right;
204         got_smaller = 1;
205     }
206     else if( descend( &(*rootp)->left, &dp ) )
207     {
208         got_smaller = balance_1( rootp );
209     }
210
211     free( dp );
212 }
213
214 return got_smaller;
215 }
216
217 /* ----- */
218
219 int delete( rootp, key, cmp )
220 HEADER **rootp;
221 HEADER *key;
222 int (*cmp) ();
223 {
224     /* Cmp is a comparison routine called with (*cmp)(key, node);
225      * where "key" is the second argument to delete and "node"
226      * is a pointer to one node in the tree. It should return
227      * <0 if key<node 0 if key==node >0 if key>node. Returns
228      * 1 if the node was deleted, 0 if the node wasn't in the
229      * tree.
230      */
231
232     Cmp      = cmp;
233     Key      = key;
234     Notfound = 0;
235
236     del( rootp );
237
238     return !Notfound;
239 }
240

```

End Listing Eight

(Listing Nine begins on next page)

MODULA-2

Modula-2 is replacing C as the premier systems-programming language for the same reason that interchangeable parts replaced craftwork in the 19th century.

REPETOIRE, from PMI, provides the parts you need to build the software of tomorrow.

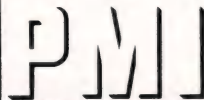
Low-Level Tools.

REPETOIRE adds full support for LONGINTs to Logitech's Modula 2/86. It also provides (1) improved DOS/BIOS functions; (2) fast byte-moves/scans; (3) better control over speaker, clock, screen, keyboard, etc.; and (4) extensive string- and list-handling tools.

High-Level Tools.

REPETOIRE includes full source for 3 high-level subsystems: (1) an unusually sophisticated **screen design/display system**: instant display of full-color screens that automatically obtain and check input, provide help, control program branching, and intelligently adapt to the hardware; (2) a **natural-language analysis system**; and (3) an integrated editor that can operate concurrently in multiple processes and in windows of any size or shape.

This is 380K of Modula-2 source code and a 200-page manual for \$64. To help you decide, we'll send you the full manual and a demo for **FREE**. For IBM compatibles; Logitech & ITC versions (ask about others). Check/MC/Visa. Call (503) 777-8844 (24 hours).



4536 S.E. 50th • Portland, OR 97206
MCI Mail: PMI • Compuserve: 74706,262

Circle no. 239 on reader service card.

the HD test

Professional Test/Format Program For Hard Drives in PC/XT/AT

- Setup interleave, step rate, etc.
- Surface analysis to flag bad tracks
- Load/save setup and bad track files
- Now supports AUTOCONFIG!
- Menu driven, with help windows
- Free PARK program included!
- Order HDTST today for only \$99!

Proto PC inc.
612-644-4660

2424 Territorial Road, St. Paul, MN 55114
Telex 910-380-7623

Circle no. 295 on reader service card.

C CHEST

Listing Nine (Listing continued, text begins on page 20.)

Listing 9 -- makefile for avl.lib

```
-----
#
#   Make avl.lib and test.exe using the Microsoft C Compiler, ver. 3.0
#   and Polymake.
#
.c.obj:
    cl -c $*.c >>err

OBJECTS = avldel.obj avlfind.obj avlfree.obj avlins.obj avlprnt.obj

# .....

test.exe: test.obj tree.lib
    cl test.obj -link tree.lib

test.obj: tree.h

# .....

tree.lib: $(OBJECTS)
    del tree.lib
    lib <@<

tree
Y
$(OBJECTS)
tools.ndx
< >>err

# .....

avlde1.obj:    avl.h
avlins.obj:    avl.h
avlfind.obj:   avl.h
avlprnt.obj:   avl.h
avlfree.obj:   avl.h
```

End Listings

IQCLISP

THE CLOSEST THING TO COMMON LISP AVAILABLE FOR YOUR PC

RICH SET OF DATA TYPES

Bignums, for high precision arithmetic
8087 support, for fast floating point
Arrays, for multidimensional data
Streams, for device-independent i/o
Packages, for partitioning large systems
Characters, strings, bit-arrays

FULL SET OF CONTROL PRIMITIVES

flet, labels, macrolet, for local functions
if, when, unless, case, cond, for conditionals
Keyword parameters, for flexibility
Multiple-valued functions, for clarity
Flavors, for object-oriented programming
Stacks, for coroutines
Closures, for encapsulation

LARGE COMPLEMENT OF FUNCTIONS

Mappers, for functional programming
format, for output control
sort, for user-specified predicates
Transcendental floating point functions
String handling functions
Over 400 functions altogether

APPLICATION SUPPORT

Save and restore full environments
User-specified initializations
Assembly language interface

HARDWARE REQUIREMENTS

8088 or 8086 CPU, MSDOS Operating System
390K RAM or more

IQCLISP

5¼" diskettes
and manual **\$300.00**
Foreign orders add \$30.00 for airmail.
U.S. Shipping included for prepaid orders.

Integral Quality
P.O. Box 31970
Seattle, Washington 98103
(206) 527-2918

Washington State residents add sales tax.
VISA and MASTERCARD accepted.

EXTENDABILITY

defstruct, to add data types
Macros, to add control constructs
Read macros, to extend the input syntax
Extendable arithmetic system
Customizable window system

DEBUGGING SUPPORT

step, for single-stepping
trace, for monitoring
break, for probing
inspect, for exploring
Flexible error recovery
Customizable pretty-printer

MSDOS INTERFACE

Random access files
Hierarchical directory access
MSDOS calls

DOCUMENTATION

On-line documentation of functions
apropos
300-page indexed reference manual

THINK

FAST!

**If you're a C programmer
you could be a more productive C programmer.**

**Introducing Lightspeed C™ for the Macintosh™
from THINK Technologies, Inc.**

Lightspeed C is a compiled programming environment for the Macintosh™ that gives you speed, convenience, and top quality code generation, too.

With Lightspeed C, turnaround is 1000% faster. Time to build from scratch is 3 times faster. Time to link a typical 15,000 line program is 5 seconds! And generated code quality is better than any on the market.

Best of all, Lightspeed C's, integrated Edit-AutoMake-Launch environment makes turnaround a one-step process.

If you want to produce higher quality results with less time and effort, send for Lightspeed C today.

The above statements are based upon benchmarks for creating an executable version of XLISP v 1.4 (16.5K source lines) from scratch and by modifying, re-compiling, and re-linking one source file. Comparisons were performed using a 512K Macintosh with a 10MB Hyperdrive.™

Circle no. 234 on reader service card.

Generated code size (in bytes)
Program build time (in secs.)

a. compile
b. link-to-run
c. TOTAL pgm build
Turnaround time (in secs.)
(time to make a change to module xlcont.c)

Consulair (MacC V4.0)	Aztec (V1.06G)	Megamax (V2.1)	Lightspeed (V0.40)
36770	34566	37698	33870
887	654	354	194
99	49	95	5
986	703	449	199
159	108	127	9

☐ **Send me Lightspeed C™
fast. \$175.00 for each
non-copy protected compiler.**

NAME

TITLE

COMPANY

ADDRESS

CITY

STATE

ZIP

TELEPHONE

☐ CHECK ENCLOSED

☐ MC ☐ VISA ☐ AMEX ACCT. #

EXP.
DATE

SIGNATURE

Mail to:

THINK Technologies
420 Bedford Street
Lexington, MA 02173
Or call 617-863-5595

C C COMPILERS

Listing One (Text begins on page 30.)

```

/*
#define EACHLOOP
*/
/*
#define VERIFY
*/

struct bmttype
{
    int          (*func) ( );
    char         *pf;
    int          loop, time;
};

```

End Listing One

Listing Two

```

bm_main.c

#include <stdio.h>
#include "bench.h"

extern struct bmttype bm[ ];

main( argc, argv )
    int      argc;
    char     *argv[ ];
{
    register struct bmttype *bmp;
    int total;
    FILE *table, *fopen();

    total = 0;

    /* Run all the benchmark
    for ( bmp = &bm[0]; bmp->loop; bmp++ )
    {
        total += (bmp->time = (*bmp->func)( bmp->loop ));
        printf( "%8.8s %5d %4d.%1d\n", bmp->pf, bmp->loop, bmp->time / 10,
        bmp->time % 10 );
    }

    /* Print out all the results

    table = fopen( "result.tbl", "a" );

```

ASMLIB

Assembly Language Programming Library for the IBM PC/XT/AT or compatible DOS systems.

ASMLIB gives the Assembly Language programmer 190+ assembly functions which do -

- Graphics functions draw CIRCLES, ARCS, ELLIPSES, LINES, and plots POINTS on the Color Adapter, Enhanced Graphic Adapter, and the Herc Monochrome Card. Functions also allow PAINTING, IMAGE SAVES and RESTORES, and SCROLLING.
- Text Windows - Up to 64 text windows may be defined, outlined, overlapped, moved, and can be grouped onto 256 logical display pages.
- Floating Point - Arithmetic and Trigonometry functions for the MS and IEEE (8087) floating point formats (both 4 and 8 byte precision), and the 8087 (80287) can be utilized automatically if installed into the target system.
- ASCII String/Numeric conversion routines provide a user interface to the math functions. ASFORMAT function allows numeric values to be formatted utilizing commas, dollar signs, left or right justified, etc. (i.e. BASIC's PRINT USING).
- Mouse Support - ASMLIB provides support for any mouse device which adheres to the MS Mouse software standard.
- Dynamic Memory support can utilize all available memory (up to 640k). Blocks of memory can be allocated, changed, moved, and killed.
- Console I/O, Disk I/O with file copy routines, Asynchronous Communications, Printer Support, ASCII String support, Sound generation, plus much more.

ASMLIB is supplied with complete MASM source code on 3 DOS diskettes along with a 215+ page reference manual.

**- ALL FOR ONLY -
\$149.00**

For ordering or info please call:
BC Associates
13073 Springdale St., Suite 134
Westminster, CA 92683
(714) 741-3015

Phone COD orders accepted - ORDER YOURS TODAY!!!

Circle no. 182 on reader service card.

C CODE FOR THE PC

source code, of course

QC88 C Compiler	\$90
Concurrent C	\$45
Coder's Prolog in C	\$45
LEX	\$25
YACC & PREP	\$25
Small-C compiler for 8088	\$20
tiny-c interpreter & shell	\$20
Xlisp 1.5a & tiny-Prolog	\$20
C Tools	\$15

The Austin Code Works
11100 Leafwood Lane
Austin, Texas 78750-3409
(512) 258-0785

Free shipping on prepaid orders

No credit cards

Circle no. 250 on reader service card.

End Listing Two

(continued on next page)



Listing Three (Listing continued, text begins on page 30.)

```

        for ( inner = 1000; inner; inner-- )
        {
/* Multiple assignments */
        i1 = i2 = i3 = i4 = i5 = 0;
        i1 = i2 = i3 = i4 = i5 = 0;
        i1 = 1; i1 = 1; i1 = 1; i1 = 1; i1 = 1; i1 = 1;
        i1 = 1; i1 = 1; i1 = 1; i1 = 1; i1 = 1; i1 = 1;
/*
Increment and Decrement */
        i1 += 1; i1 += 1; i1 += 1; i1 += 1; i1 += 1;
        i2 -= 1; i1 += 1; i1 += 1; i1 += 1; i1 += 1;
/*
Multiply by two (left shift) */
        i1 = 1;
        i1 *= 2; i1 *= 2; i1 *= 2; i1 *= 2; i1 *= 2;
/* Compile
time expression eval */
        i1 = (10 * (27 + 14 + 1) - 1) * 47 + 32;
/* Compare equivalent constructions */
        i1 += 1;
        i1 = i + 1;
        i1++;
/* Common
subexpressions */
        i2 = (i4 * i5) + i3;
        i1 = (i4 * i5) + i2;
        i3 = ((i4 * i5) + i2) * ((i4 * i5) + i1);
/* Code motion:
extraction from loop */
        for (i1 = i2; i1; i1--)
        {
            i3 = array[ inner + i2 ];
            i4 = array[ inner + i1 ];
            i5 = array[ i3 + 1 ];
            i4 = array[ i2 + 1 ];
            i5 = array[ i2 + 1 ];
        }
    }
    return( time_n( ) );
}

```

End Listing Three

Parallel Programming for "C"

INTERWORK

(formerly Teamwork)

A Concurrent Programming Toolkit

Interwork is a "C" program library which allows you to write your programs as a set of cooperating concurrent tasks. Very useful for simulation, real-time applications, and experimentation with parallel programming.

FEATURES

- Supports a very large number of tasks (typically more than 100) limited only by available memory. Low overhead per task results in very fast context switching.
- Provides a full set of inter-task communication (ITC) facilities, including shared memory, locks, semaphores, blocking queues, and UNIX*-style signals. Also has building blocks for constructing your own ITC facilities.
- Handles interrupts and integrates them into task scheduling. Supply your own interrupt handlers or block tasks on interrupts.
- Lets you trace task switches and inter-task communication.
- Comes with complete documentation including a user's manual and reference manual of commands.

Interwork is available for the following systems:

Hardware	Operating System	Price
IBM PC, XT, AT	PC-DOS 2.0 or later	\$ 65
IBM PC AT	XENIX*	\$ 85
DEC VAX*	UNIX 4.2BSD	\$195

PC-DOS version is compatible with DeSmet, Lattice, and Microsoft C compilers.

Please specify hardware and operating system when ordering. Shipping and handling included. Send check or money order to:



Block Island Technologies
Innovative Computer Software

13563 NW Cornell Road, Suite 230, Portland, Oregon 97229-5892
(503) 241-8971

*Trademarks: UNIX, AT&T Bell Laboratories, Inc.; XENIX, Microsoft, Inc.; VAX, Digital Equipment Corporation

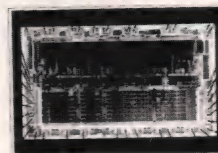
Circle no. 275 on reader service card.

FROM THE DEVELOPERS OF THE

65816

The programming handbook you've been waiting for...

Programming the
65816 Microprocessor
Including 6502 and 65C02



Brady

David Eyes

0-89303-789-3/288 p/\$22.95

- Outlines the programming strengths of the 65816, 6502, and 65C02.

- Includes a review of basic concepts, architecture, and logical operations.

Now available at your local bookstore, or call toll free 1-800-624-0023 to order your copy today. In New Jersey, call 1-800-624-0024.

Brady

Circle no. 219 on reader service card.

Dr. Dobb's Journal, August 1986

Listing Four

```
pointer.c

#include "bench.h"

extern int pointer();

struct bmttype bm[] =
{
    {
        pointer, "pointer", 1500, 0
    },
    /* End of list */
    {
        pointer, "", 0, 0
    }
};

static int s[10][10][10];
static int d[10][10][10];

pointer( loop )
/*
    * This benchmark copies one three dimensional
    * array to another using pointers. It is
    * complicated by the use of three levels of
    * pointer indirection to increase the ratio of
    * pointer arithmetic to overhead.
    */

{
    int loop;

    int *sp1, **sp2;
    int *dp1, **dp2;
    register int ***sp3, ***dp3;

    sp2 = &sp1;
    sp3 = &sp2;
    dp2 = &dp1;
    dp3 = &dp2;

    time_0();

    for ( ; loop; loop-- )
    {
```

(continued on next page)

THE HAMMER Software Tools in C

"I have already saved weeks of coding ... thank you for providing such a useful tool ..." - G.T.

Let **The HAMMER Library** of over 150 routines save you valuable development time and effort:

LIBRARY FUNCTIONS

- **Multi-level 123-like MENUS**
- **DATA ENTRY**
 - MULTI-FIELD mode for Full-Screen data entry
 - Single-Field mode for individual fields
 - Data Verification
 - Full Editing within each field
 - Strings, dates, and fixed decimal numbers
 - "Option" fields force user to pick from a given set
- **SCREEN MANAGEMENT**
 - cursor positioning
 - display boxes & tables
 - full attribute control
 - scrolling and clearing
- Date/time/string conversions
- BIOS access/pattern matching/and more

UTILITIES

- HARC - complete Source File Archiver
- HAMCC - compile designated source modules residing **WITHIN an archive file** under any of the supported compilers and optionally place resulting object modules in a library.

SUPPORTED C COMPILERS:

Microsoft C 3.00 • CI-C86 • Mark Williams C86
DeSmet C • Lattice

INCLUDES source code and manual \$195 plus shipping
VISA/MC accepted

O.E.S. SYSTEMS

1906 Brushcliff Rd. • Pittsburgh, PA 15221 • 412/243-7365

Looking for the right tool for the job?

REACH FOR THE HAMMER

Circle no. 137 on reader service card.

Improve Program Quality Enhance Program Productivity

Design your programs around ...

ASE, the Aspen Systems Subroutine Editor you can call from your programs. With **ASE** you can easily:

- ★ Design you own screen layouts for Program Input
- ★ Color and/or highlight input fields
- ★ Define your own key functions
- ★ Convert and edit a wide variety of fields
- ★ Update in several windows simultaneously

ASE includes 2 major subroutines, many minor subroutines and install and demonstration programs. Data & screen layouts described in a single map.

Price **\$99**

Available MSDOS 1.2,3

Demo Disk **\$5**

ASP, the Aspen Systems Subroutine Package provides functions difficult or unavailable in some higher level languages, performs those available with greater speed/ease or smaller memory requirements.

The **ASP** Package includes:

- ★ 100+ subroutines
- ★ A 300 page manual packed with examples
- ★ Test, Demonstration and customization source

Price **\$130**

Available MSDOS 1.2,3, CP/M

All subroutines are callable from assemblers and Microsoft Compilers, easily altered for other compilers, can be used in EXE or COM programs.

Prices are PPD (continental USA).
Colorado Residents add 3%.

P. O. Box 1163
Grand Junction, CO 81502
(303) 245-3262
VISA/MasterCard accepted

**ASPEN
SYSTEMS**

CP/M and MS-DOS are trademarks of
Digital Research and Microsoft, respectively.

Circle no. 277 on reader service card.

Listing Four (Listing continued, text begins on page 30.)

```

        dpl = &d[0][0][0];
        for ( spl = &s[0][0][0]; spl <= &s[9][9][9]; spl++, dpl++ )
            ***dp3 = ***sp3;
    }

    return( time_n() );
}

```

End Listing Four

Listing Five

```

trig.c

#include "bench.h"

extern int trig( );

struct btype bm[ ] =
{
    {
        trig, "trig", 100, 0
    },
    {
        /* End of list */
        trig, "", 0, 0
    }
};

int trig( loop )
/*
 * This benchmark function tests the simple
 * trigonometric functions sin, cos, and tan.
 * It does 12 of each operation, and assumes
 * radian arguments.
 */
{
    int loop;
}

```

BLAST[®]



PC-MINI-MAINFRAME COMMUNICATIONS SOFTWARE

ANY COMPUTER WITH BLAST CAN TALK TO ANY OTHER COMPUTER WITH BLAST, the universal file transfer utility linking many different computers, operating systems, and networks, via RS 232 serial ports.

NO ADD-ON BOARDS TO BUY! BLAST software uses any asynchronous modems or direct connect for fast, error-free data transfer through noisy lines and PBXs, across LANs, and over satellites or packet switched networks.

THE PERFECT LOW-COST LINK FOR PC's, MINIS, MAINFRAMES Transfer binary or text files, or executable commands. Use BLAST standalone, or built it into your application.

\$250/Micros \$495-895/Minis \$2495/up Mainframes

COMMUNICATIONS RESEARCH GROUP (800)-24-BLAST

8939 Jefferson Hwy. Baton Rouge, LA 70809 (504)-923-0888

Circle no. 272 on reader service card.

PC/VI

Full Screen Editor for MS-DOS (PC-DOS)

Looking for an Ultra-Powerful Full-Screen editor for your MS-DOS or PC-DOS system? Are you looking for an editor **FULLY COMPATIBLE** with the UNIX* VI editor. Are you looking for an editor which not only runs on IBM-PC's and compatibles, but **ANY** MS-DOS system? Are you looking for an editor which provides power and flexibility for both programming and text editing? If you are, then look no further because **PC/VI IS HERE!**

The following is only a hint of the power behind **PC/VI**: English-like syntax is command mode, mnemonic control sequences in visual mode; full undo capability; deletions, changes and cursor positioning on character, word, line, sentence, paragraph or global basis; editing of files larger than available memory; powerful pattern matching capability for searches and substitutions; location marking; joining multiple lines; auto-indentation; word abbreviations and **MUCH, MUCH MORE!**

The **PC/VI** editor is available for IBM-PC's and generic MS-DOS based systems for only \$149. For more information call or write:

Custom Software Systems
P.O. Box 678
Natick, MA 01760
617-653-2555

The UNIX community has been using the VI editor for years. Now you can run an implementation of the same editor under MS-DOS. Don't miss out on the power of **PC/VI!**

*UNIX is a trademark of AT&T Bell Laboratories.

Circle no. 268 on reader service card.


```

{
    double a, b, c, d;
    double sin( ), cos( ), tan( );

    time_0( );

    for ( ; loop; loop--)
    {
        a = sin( .392699 );
        b = sin( .785398 );
        c = sin( 1.178097 );
        d = sin( 1.963495 );
        a = sin( 2.356194 );
        b = sin( 2.748893 );
        c = sin( 3.534292 );
        d = sin( 3.926991 );
        a = sin( 4.319690 );
        b = sin( 5.105088 );
        c = sin( 5.497787 );
        d = sin( 5.890486 );
        a = cos( .392699 );
        b = cos( .785398 );
        c = cos( 1.178097 );
        d = cos( 1.963495 );
        a = cos( 2.356194 );
        b = cos( 2.748893 );
        c = cos( 3.534292 );
        d = cos( 3.926991 );
        a = cos( 4.319690 );
        b = cos( 5.105088 );
        c = cos( 5.497787 );
        d = cos( 5.890486 );
        a = tan( .392699 );
        b = tan( .785398 );
        c = tan( 1.178097 );
        d = tan( 1.963495 );
        a = tan( 2.356194 );
        b = tan( 2.748893 );
        c = tan( 3.534292 );
        d = tan( 3.926991 );
        a = tan( 4.319690 );
        b = tan( 5.105088 );
        c = tan( 5.497787 );
        d = tan( 5.890486 );
    }

    return( time_n( ) );
}

```

End Listings

BACK ISSUES

1982	1983	1984	1985	1986
#68—June	#77—March	#87—Jan.	#99—Jan.	#113—March
#69—July	#78—April	#88—Feb.	#104—June	#114—April
#70—Aug.	#79—May	#89—March	#108—Oct.	#115—May
#71—Sept.	#80—June	#90—April	#109—Nov.	#116—June
#72—Oct.	#81—July	#91—May	#110—Dec.	#117—July
#73—Nov.	#82—Aug.	#92—June		
	83—Sept.	#93—July		
	#84—Oct.	#94—Aug.		
	#85—Nov.	#95—Sept.		
	#86—Dec.	#96—Oct.		
		#97—Nov.		

TO ORDER: send \$5 for 1 issue, \$4.50 each for 2-5, \$4 each for 6 or more to: Dr. Dobb's Journal, 501 Galveston Drive, Redwood City, CA 94063

Name _____

Address _____

City _____

State _____

Zip _____

3118

C Cross Compiler

68000/08/10/20

Features:

- Full, Standard C
- Easy to Use Compiler Options
- Complete User Documentation
- Global Code Optimization
- Optional Register Allocation Via Coloring
- ROMable and Reentrant Code
- Comprehensive Royalty Free Run-time Library
- Floating Point Library Routines
- Intermix MCC68K C with ASM68K Assembly Language or Microtec PAS68K Pascal
- Optional Assembly Language Listing Intermixed with MCC68K C Source Line Number
- Symbolic Debug Capability

The Microtec MCC68K C Cross Compiler is a complete implementation of the "C" programming language as defined in The C Programming Language by Kernighan and Ritchie with extensions.

MCC68K emits highly optimized assembly language code for the Microtec ASM68K Motorola compatible assembler.

The Microtec MCC68K package includes the compiler, relocatable macro assembler, linking loader, run-time library, and comprehensive user's guide.

Now Generates:
Position Independent Code

Host computers include: DEC VAX, DG MV-Series, Apollo, IBM PC and PC-compatibles..

We're **Functional** and **Fast** and **Serious** about our products. We've been providing flexible and economical solutions for software developers since 1974.

Beginning with product concept, through development, quality assurance, and post-sales support - **Quality, Compatibility and Service** are the differences which set Microtec Research apart from others.

If you're a serious software developer, shopping for software development tools, call or write today for more information:

800-551-5554,
In CA call (408) 733-2919.

3930 Freedom Circle, Suite 101, Santa Clara, CA 95054
Mailing Address: P.O. Box 60337, Sunnyvale, CA 94088

MICROTEC[®]
RESEARCH

STRUCTURED PROGRAMMING

Listing One (Text begins on page 116.)

Listing One. Ada procedure to swap two integers.

```
procedure Swap(First, Second : in out integer) is
  Temporary : integer;
begin
  Temporary := First;
  First := Second;
  Second := Temporary;
end Swap;
```

End Listing One

Listing Two

Listing Two. Generic Ada procedure to swap two scalars.

```
generic
  -- Declare generic types here
  type Object is private;
  -- List heading for generic routines here
  procedure Swap(First, Second : in out Object);
-- Full definition of procedures is below
procedure Swap(First, Second : in out Object) is
  Temporary : Object;
begin
  Temporary := First;
  First := Second;
  Second := Temporary;
end Swap;
```

End Listing Two

Listing Three

Listing Three. Generic Ada procedure to return the next element in a circular list.

```
generic
  type Circular_Item is (<>);
  function Fetch_Next_In_Circular_List(Member : Circular_Item)
    return Circular_Item;
-- Declare the generic function body
function Fetch_Next_In_Circular_List(Member : Circular_Item)
  return Circular_Item is
begin
  -- use predefined LAST attribute
  if Member = Circular_Item'LAST
  then -- use predefined FIRST attribute
    return Circular_Item'FIRST;
  else -- use predefined SUCCEsive attribute
    return Circular_Item'SUCC(Member);
  end if;
end Fetch_Next_In_Circular_List;

-- Examples for generic instantiation are
-- type Day is (MON, TUE, WED, THU, FRI, SAT, SUN);
-- function NextDay is new Fetch_Next_In_Circular_List(Day);
-- NextDay(TUE) returns WED
-- NextDay(SUN) returns MON

-- subtype Hours is integer 0..24;
-- function NextTime is new Fetch_Next_In_Circular_List(Hours);
-- NextTime(4) returns 5
-- NextTime(24) returns 0
```

End Listing Three

Listing Four

Listing Four. Generic Ada function that scans an array and returns the largest value found.

```
generic
  type Index_Range is range <>;
  type Member is range <>;
```

```
  type List is array (Index_Range) of Member;
  function Largest(L : List) return Member;

function Largest(L : List) return Member is
-- Initialize Big to lowest value
Big : Member := Member'FIRST;

begin
  for i in Index_Range loop
    if Big < L(i) then Big := L(i); end if;
  end loop;
  return Big;
end Largest;
```

End Listing Four

Listing Five

Listing Five. Generic Ada function to return the average of a floating point typed array.

```
generic
  type Index_Range is range <>;
  type Element is digits <>;
  type List is array (Index_Range) of Element;
  function Average(X : List) return Element;

function Average(X : List) return Element is
  Sum : Element := 0.0; -- Initialize summation

begin
  for i in Index_Range loop
    Sum := Sum + X(i);
  end loop;
  return (Sum / FLOAT(Index_Range));
end Average;
```

End Listing Five

Listing Six

Listing Six. Generic Ada procedure to solve the mathematical root of a function.

```
generic
  type Floating is digits <>;
  -- declaring a subprogram parameter
  -- the "with" keyword distinguishes it from other
  -- declared generic routines.
  with function F_of_X(X : Floating) return Floating;
  procedure Root(Guess : in out Floating; Accuracy : in Floating;
    Iter_Max : in INTEGER; Converge : out BOOLEAN);

  procedure Root(Guess : in out Floating; Accuracy : in Floating;
    Iter_Max : in INTEGER; Converge : out BOOLEAN) is
    Increment, Diff : Floating;
    Iter : INTEGER := 0;

  begin
    Converge := true;
    loop
      if abs(Guess) > 1.0
      then Increment := 0.01 * Guess;
      else Increment := 0.01;
      end if;
      Diff := 2.0 * Increment * F_of_X(Guess) /
        (F_of_X(Guess + Increment) -
         F_of_X(Guess - Increment));
      Guess := Guess - Diff;
      Iter := Iter + 1;
      if Iter > Iter_Max then Converge := false; end if;
      if (abs(Diff) < Accuracy) or (not Converge)
      then exit;
      end if;
    end loop;
  end Root;
```

End Listing Six

Listing Seven

Listing Seven. Generic Shell sort procedure in Ada.

```
generic
  type Range_Index is (<>);
  type Data is private;
  type List is array (Range_Index range <>) of Data;
```



```

-- declare generic function/operator
with function ">" (A,B : Data) return BOOLEAN;
procedure Shell_Sort(L : in out List; Num : INTEGER);

procedure Shell_Sort(L : in out List; Num : INTEGER) is

Offset, I, K : INTEGER;
Tempo : Data;
In_Order : BOOLEAN;

begin
  Offset := Num;
  while Offset > 1 loop
    Offset := Offset / 2;
    loop
      In_Order := true;
      K := Num - Offset;
      for J in 1..K loop
        I := J + Offset;
        if L(J) > L(I) -- Using the ">" operator
          then In_Order := false;
              Tempo := L(I);
              L(I) := L(J);
              L(J) := Tempo;
            end if;
      end loop;
      if In_Order then exit; end if;
    end loop; -- open loop
  end loop; -- while loop
end Shell_Sort;

```

End Listing Seven

Listing Eight

Listing Eight. Generic Modula-2 function to search for a specific value in an integer/cardinal array.

```

PROCEDURE LinearSearch (VAR Element : ARRAY OF WORD; (* input *)
                        SearchValue : INTEGER; (* input *)
                        VAR Index : CARDINAL (* output *)
                        ) : BOOLEAN;

```

```

VAR Found : BOOLEAN;
hi : CARDINAL;

```

```

BEGIN
  Index := 0; hi := HIGH(Element); Found := FALSE;
  WHILE (Index <= hi) AND (NOT Found) DO
    (* Logical expression tested converts *)
    (* array element into an integer type *)
    IF SearchValue = INTEGER(Element[Index])
      THEN Found := TRUE
      ELSE INC(Index)
    END; (* IF *)
  END; (* WHILE *)
  RETURN Found
END LinearSearch;

```

End Listing Eight

Listing Nine

Listing Nine. Generic Modula-2 Shell sort procedure.

```

PROCEDURE ShellSort (VAR L : ARRAY OF WORD; (* in/out *)
                     Sample1,
                     Sample2 : ARRAY OF WORD; (* input *)
                     Num : CARDINAL; (* input *)
                     IsGreater : UserDefinedProc; (* input *)

```

```

VAR Offset, I, K, DataSize : CARDINAL;
In_Order : BOOLEAN;

```

```

PROCEDURE FetchItem (Item Num : CARDINAL; (* input *)
                     VAR Item : ARRAY OF WORD; (* output *)
                     (* Procedure copies an element from main array in Item *)

```

```

VAR Count : CARDINAL;

```

```

BEGIN
  FOR Count := 0 TO DataSize - 1 DO
    Item[Count] := L[Count + Item_Num * DataSize]
  END;
END FetchItem;

```

```

PROCEDURE PutItem (Item Num : CARDINAL; (* input *)
                  VAR Item : ARRAY OF WORD; (* output *)

```

(continued on next page)

**NOT COPY
PROTECTED!**



Sybil Is an Advanced Diagnostics disk ...

She can low format hard disks just like Advanced Diagnostics (IBM, Compaq, etc.) and she can do system and memory tests which provide even more information than Advanced Diagnostics does. \$245.00 cheaper than IBM's Advanced Diagnostics!

Sybil Is a Disaster Recovery program ...

She can recover hard disks that have been accidentally formatted, completely! The hard disk reappears in exactly the same condition prior to the format. Truly amazing!

Sybil Is a Graphics Editor ...

She can draw on either RGB monitors (in color) or IBM Monochrome monitors in high ASCII characters. Perfect for creating Binary Image Files. The Binary Image Files can be converted to Assembly and then linked to other languages, such as your favorite Pascal, C, or compiled BASIC program. Includes source code.

Sybil Is a File Wizard ...

Sybil can backup files by **date**, by **time**, or by **size**. She can find any file (or files) anywhere on your hard or floppy disks, even if you haven't the vaguest notion. She can edit file attributes with the greatest of ease, unerase files, edit sectors, and globally change time and date stamps. All her file utilities understand paths and wildcards.

Sybil Is also a ...

RAM Disk, Print Spooler, General Regular Expression Parser and, Advanced File Comparator.

Order Sybil Today!

**Call 800-922-3001, in Colorado,
303-444-1542**



SOPHCO

**PO Box 7430
Boulder, Colorado 80306**



STRUCTURED PROGRAMMING

Listing Nine (Listing continued, text begins on page 116.)

```
(* Procedure copies an element to main array *)
VAR Count : CARDINAL;

BEGIN
  FOR Count := 0 TO DataSize - 1 DO
    L[Count + Item_Num * DataSize] := Item[Count]
  END;
END PutItem;

BEGIN (* ----- Shell Sort ----- *)
  DataSize := HIGH(Sample1) + 1;
  Offset := Num;
  WHILE Offset > 1 DO
    Offset := Offset DIV 2;
    REPEAT
      In Order := TRUE;
      K := Num - 1 - Offset;
      FOR J := 0 TO K DO
        I := J + Offset;
        FetchItem(I, Sample1);
        FetchItem(J, Sample2);
        (* Logical expression employs *)
        (* user-supplied logical function *)
        IF IsGreater(Sample1, Sample2)
          THEN In Order := FALSE;
              (* Swap items *)
              PutItem(J, Sample1);
              PutItem(I, Sample2);
            END; (* IF *)
      END; (* FOR *)
    UNTIL In Order;
  END; (* WHILE *)
END Shell_Sort;
```

End Listing Nine

Listing Ten

Listing TEN. Modula-2 function compares "frequency" fields.

```
PROCEDURE GreaterFreq(Field1, Field2 : ARRAY OF WORD) : BOOLEAN;

VAR Ptr1, Ptr2 : POINTER TO NameUse; (* record type defined *)
                                     (* elsewhere in program *)

BEGIN
  (* Get address of records *)
  RecordPointer1 := ADR(Field1);
  RecordPointer2 := ADR(Field2);
  RETURN RecordPointer1^.Frequency > RecordPointer2^.Frequency
END GreaterFreq;
```

End Listing Ten

Listing Eleven

Listing Eleven. Iterator example. Professional Pascal program compares a list of names with a list of keys and report any matches found.

```
program Pick_Data;

const MAX_NAME = 1000;
      MAX_KEY = 50;

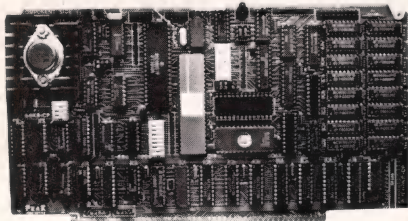
type Name_type = String(80);
      Name_Array = array [1..MAX_NAME] of Name_type;
      Key_Array = array [1..MAX_KEY] of Name_type;
      Count = array [1..MAX_KEY] of Integer;

var K : Integer;
      Names : Name_Array;
      Keys : Key_Array;
      Key_Count : Count;
      Num_Name, Num_Key : Integer;
      Name_File, Key_File : Text;

iterator Select (Num_Name, Num_Key) :
  (Key_Index, Name_Index : Integer);
var I, J : Integer;
begin
  (* Loop counters are automatic in Prof. Pascal *)
  for I := 1 to Num_Key do
    for J := 1 to Num_Name do
      if Keys[J] = Names[I]
      then begin
        Key_Count[J] := Key_Count[J] + 1;
        Yield(J, I)
      end
    end
  end;

begin
  Reset(Name_File, 'NAMES.TXT'); Num_Name := 0;
  Reset(Key_File, 'KEYS.TXT'); Num_Key := 0;
  (* Read names from name file *)
  while not EOF(Name_File) do begin
    Num_Name := Num_Name + 1;
    Readln(Name_File, Names[Num_Name]);
  end;
  Close(Name_File);
  (* Read keys from name file *)
  while not EOF(Key_File) do begin
    Num_Key := Num_Key + 1;
    Key_Count[Num_Key] := 0;
    Readln(Key_File, Keys[Num_Key]);
  end;
  Close(Key_File);
  (* Loop that finds and displays matching keys and names *)
  for Key_Index, Name_Index in Select (Num_Name, Num_Key) do
    Writeln(Keys[Key_Index], 'is key # ', Key_Index,
            ' matches name # ', Name_Index);
  end;
  (* Loop to display name matching frequency *)
  for K := 1 to Num_Key do
    Writeln('Key # ', K, ' has found ', Key_Count, ' matched
            names');
  end.
```

End Listings



68K8-CP

Expand Your System with a 68000 CoProcessor

Peak Electronics' 68K8-CP is a high performance 68000 software development package designed to easily integrate into your existing S-100 system. The package consists of the 68K8-CP coprocessor card, CP/M-68K, and a software toolkit that includes a UNIX V7 compatible floating point C compiler and a symbolic debugger.

Any system running CP/M®-2.2, CP/M-3.0 or CP/M-86 can be running CP/M-68K within minutes without any change in existing hardware or software. This card does not replace your current processor. All of the original system's devices (RAM, disks, and other peripherals) are immediately available to the user of CP/M-68K. All files can be accessed by whichever operating system is currently active. Control is transferred between operating systems with a simple one line command.

Features:

- Does not replace your current CPU card or software
- Includes CP/M-68K with UNIX® V7 compatible floating point C compiler and a symbolic debugger
- All developed C and Assembly code is fully relocatable and ROMable
- 8 or 10Mhz CPU with no wait state RAM
- 128K bytes of RAM expandable to 512K
- 2 serial and 1 parallel I/O ports
- IEEE-696-1983, S-100 Compatible
- 30 day money back guarantee
- 1 year parts and labor warranty

Complete Package: \$995.00
VISA or Master Card Accepted

PEAK
Electronics
P.O. Box 700112, San Jose, CA 95170-0112
(408) 253-5108

MS-DOS, UNIX, APPLE MAC, CP/M, NETWORKS and MORE. ONE c-tree ISAM DOES THEM ALL!

The creator of Access Manager™ brings you the most powerful C source code, B+ Tree file handler: **c-tree™**

- multi-key ISAM and low-level B+ Tree routines
- complete C source code written to K&R standards
- single-user, network and multi-tasking capabilities
- fixed and variable record length data files
- virtually opened files accommodate limited file descriptors
- no royalties on application programs

\$395 COMPLETE

Specify diskette format:

- 5¼" MS-DOS
- 8" CP/M
- 3½" Mac
- 8" RT-II



For VISA, MC and COD orders
call (314) 445-6833

FairCom

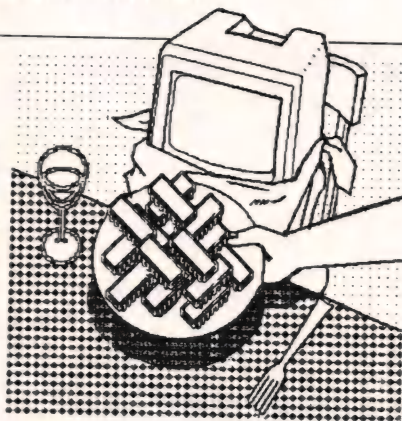
2606 Johnson Drive
Columbia, MO 65203

© 1985 FairCom

The following are trademarks: c-tree and the circular disk logo—FairCom; MS—Microsoft Inc.; CP/M and Access Manager—Digital Research Inc.; Unix—AT&T; Apple—Apple Computer Co.

Circle no. 93 on reader service card.

Fatten Your Mac



Thanks to Macintosh owners everywhere, *Dr. Dobb's* January 1985 issue #99 was a runaway best-seller.

Now, due to popular demand, the Doctor has reprinted the sought-after **Fatten Your Mac** article from the sold-out January issue. The article explains how you can pack a full 512K of memory into your system, and save half the cost by performing the upgrade yourself.

To order: Enclose \$5.00 for each copy with this coupon and send to:

Dr. Dobb's Journal, 501 Galveston Dr., Redwood City, CA 94063

Outside U.S., add \$2.00 per copy for shipping & handling.

Please send me _____ copies of **Fatten Your Mac**. **ALL REPRINT ORDERS MUST BE PREPAID.**

Name _____

Address _____

City _____ State _____ Zip _____

The 8051 Time-Saver.



**Archimedes
full power C-51 Kit.**

The ANSI-standard C
for Intel 8051 microcontrollers.

- Several memory models • Optimization options • Important microcontroller 'clib' functions like 'printf' • 32-bit IEEE floating point support • Listings and cross-references • 255-character identifiers • State-of-the-art error handling • Fast single-pass RAM-compiler • ASM source generation option • Macro-assembler • Librarian • Relocatable linker • Built in type-checking via 'LINT'-feature • Large symbol table • Standard PROM-support by Intel and Tektronix hex • Special emulator converter utilities for universal symbolic debug • Supports all 8051 proliferation chips • 30-day money-back guarantee.

Finish your 8051 projects in record-time. Program in the standard high-level language. Use old 'generic' C-code. Choose the 8051 Time-Saver. The Archimedes full power C-51.

PC or VAX?



**ARCHIMEDES
SOFTWARE**

► (415) 771-3303

Archimedes Software, Inc.
1728 Union Street
San Francisco
CA 94123

PRICES: PC at \$995.00

VAX at \$3995.00

TRADEMARKS: Archimedes: Archimedes Software, Inc.
VAX: Digital Equipment Corporation.

Circle no. 178 on reader service card.

THE PROGRAMMER'S SHOP

helps save time, money and cut frustrations. Compare, evaluate, and find products.

RECENT DISCOVERY

Quick Basic v2.0 - New user interface. Editor, source level error detection, built-in debugger, in-memory compilation, build and link user libraries. BASICA, GW-BASIC compatible. BLOAD, BSAVE PC \$ 79

AI-Expert System Dev't

Arity System - incorporate with C programs, rule & inheritance PC \$ 295
 Expertech - Powerful, no limit on memory size. Samples PC \$ 399
 EXSYS - Improved. Debug, file & external program access. MS \$ 339
 Insight 2 - dB2, language. MS \$ 879
 Others: APES (\$359), Advisor (\$949), ES Construction (\$100), ESP (\$845), Expert Choice (\$449)

AI-Lisp

BYSO - Common, MacLISP compatible. 250+ functions, fast PC \$ 150
 GC LISP Interpreter - "Common", rich. Interactive tutorial Call
 Microsoft MuLisp 85 \$ 199
 TLC LISP - classes, compiler. MS \$ 225
 TransLISP - learn fast MS \$ 75
 WALTZ LISP - "FRANZ LISP" - like, big nums, debug, CPM-80 MS \$ 149
 Others: IQ LISP (\$155), UNX LISP (\$59), IQC LISP (\$269)

AI-Prolog

ARITY Standard - full, 4 Meg Interpreter - debug, C, ASM PC \$ 350
 COMPILER/Interpreter-EXE PC \$ 795
 With Exp Sys, Screen - KIT PC \$1250
 MacProlog MAC \$ 295
 MicroProlog - enhanced MS \$ 229
 Prof. MicroProlog-full memory MS \$ 359
 Prolog-86 - Learn Fast, Standard, tutorials, samples MS \$ 95
 Prolog-86 Plus - Develop MS \$ 250
 TURBO PROLOG by Borland PC \$ 79
 Others: Prolog-I (\$365), Prolog-2 (\$1795)

AI-Other

METHODS - SMALLTALK has objects, windows, more PC \$ 79
 QNIAL - Combines APL with LISP. Source or binary. PC \$ 359

FEATURES

Microsoft Windows Software Development Kit - Run graphics programs on all machines that support Microsoft Windows. Eliminates need for hardware-specific support. Includes MS Windows user version PC \$399

Alice v1.3 - Learn Pascal. Interpreter, extensive help, automatic debugging, editor, 8087 support, graphics and windowing functions. Turbo Pascal compatible PC \$ 85

National Accounts Center

Special service is provided by a separate team for organizations with over 500 employees. Purchasing agents, evaluators, managers, and programmers all appreciate the extra information, attention to shipping and invoicing, and help finding and evaluating products. Call 800-446-1185.

Our Services:

- Programmer's Referral List
- Compare Products
- Help find a Publisher
- Evaluation Literature FREE
- BBS - 7 PM to 7 AM 617-826-4086
- Dealers Inquire
- Newsletter
- Rush Order
- Over 700 products
- National Accounts Center

Basic

ACTIVE TRACE Debugger - BASICA, MBASIC, well liked MS \$ 79
 APC MegaBASIC - powerful PC \$ 339
 Basic Development System - for BASICA; Adds Renum, more. PC \$ 105
 Basic Windows by Syscom PC \$ 95
 Better Tools - for Better Basic PC \$ 95
 CADSAM FILE SYSTEM - full ISAM in MBASIC source. MS \$ 75
 DataBurst - create screens PC \$ 119
 GoodBas - maintain code PC \$ 95
 LPI Basic - MS compatible UNIX \$1100
 MegaBasic - network support MS \$ 375
 Prof. Basic - Interactive, debug PC \$ 79
 8087 Math Support PC \$ 47
 TRUE Basic - ANSI PC \$ 119
 Run-time Module PC \$ 459

Cobol

Flexgen - report generator MS \$ 495
 LPI Cobol - ANSI '74 UNIX \$1200
 Macintosh COBOL - full MAC \$ 459
 MBP - Lev. II, native MS \$ 885
 MicroFocus Professional - full PC Call
 Microsoft Cobol Tools - xref, debugger w/source support. Xenix \$359 PC \$ 239
 Microsoft Version II - upgraded. Full Lev. II, native, screens. MS \$ 479
 Realia - very fast MS \$ 839
 Ryan McFarland COBOL MS \$ 699
 COBOL-8X MS \$1049

Editors for Programming

BRIEF Programmer's Editor - undo, windows, reconfigure PC Call
 C Screen Editor - w/source 80/86 \$ 75
 EMACS by UniPress - powerful, multifile, windows Source: \$949 \$299
 Epsilon - like EMACS, full C-like language for macros. PC \$169
 Kedit - like XEDIT PC \$109
 Lattice Screen Editor - multiwindow, multitasking Amiga \$100 MS \$109
 PMATE - power, multitask 80/86 \$149
 XTC - multitasking PC \$ 85

Atari ST & Amiga

We carry full lines of Manx, Lattice, & Metacomco.
 Amiga - LINT by Gimpel Amiga \$ 79
 Cambridge LISP Amiga \$ 200
 Lattice C ST, Amiga \$ 139
 Lattice Text Utilities Amiga \$ 75
 Megamax - tight, full ST \$ 200

RECENT DISCOVERY

Smalltalk-80 - "Official" Xerox licensed version. Compatible with Smalltalk-80 v2 running on other systems. Compiler, complete graphics interface, debugger, windows, text and graphics editor, source. Protected mode, RAM access. Requires PC AT PC \$995

C Language-Compilers

AZTEC C86 - Commercial PC \$499
 AZTEC C65 - Personal Apple II \$199
 C86 by CI - 8087, reliable MS \$299
 Lattice C - from Lifeboat MS \$289
 Lattice C - from Lattice MS \$339
 Mark Williams - w/debugger MS \$399
 Microsoft C 3.0 MS \$259
 Q/C 88 by Code Works - Compiler source, decent code, native MS \$125
 Wizard C - Lattice C compatible, full sys. III, lint, fast. MS \$389

C Language-Interpreters

C-terp by Gimpel - full K & R MS \$249
 INSTANT C - Source debug, Edit to Run-3 seconds MS \$389
 Interactive C by IMPACC Assoc. Interpreter, editor, source, debug. PC \$225
 Introducing C - self paced tutorial PC \$109
 Run/C Professional - Run/C plus create add-in libraries, more MS \$189
 Run/C Lite - improved MS \$109

C Libraries-General

Blaise C Tools 1 (\$109), C Tools 2 \$ 89
 C Essentials - 200 functions PC \$ 85
 C Food by Lattice-ask for source MS \$109
 C Utilities by Essential - Comprehensive screen graphics, strings, source. PC \$139
 C Worthy Library - Complete, machine independent, source MS \$295
 Entelekon C Function Library PC \$119
 Entelekon Superfonts for C PC \$ 45
 Greenleaf Functions - portable, ASM \$139
 PforCe by Phoenix - objects PC \$299

C Libraries-Communications

Asynch by Blaise PC \$149
 Greenleaf - full, fast PC \$139
 Software Horizons - pack 3 PC \$119

C Libraries-Files

FILES: C Index by Trio - full B + Tree, vary length field, multi compiler /File is object only MS \$ 89
 /Plus is full source MS \$349
 C to dBase - with source MS \$139
 CBTREE - sequential, source, no royalties MS \$ 99
 CTree by Faircom - no royalties MS \$339
 dbVISTA - full indexing, plus optional record types, pointers, Network. Object only - MS C, LAT, C86 \$159
 Source - Single user MS \$429
 Source - Multiuser MS \$849
 dBASE Tools for C PC \$ 79
 dbc Isam by Lattice MS \$199

We support MSDOS (not just compatibles), PC DOS, Xenix-86, CPM-80, Macintosh, Atari ST, and Amiga.

THE PROGRAMMER'S SHOP

provides complete information, advice, guarantees and every product for Microcomputer Programming.

BetterBASIC SPECIAL BUY NOW

Access full memory, program in a structured environment, utilize true procedures and functions like PASCAL, create user defined keywords like C, and access the 8087/80287 chip. GW-BASIC, PC-BASICA compatible. C-Link allows you to access your existing C Libraries from BetterBASIC.

Order before 8/31/86 and mention this ad for the special prices below:

	LIST	Before 8/31	After 8/31
BetterBASIC	199	139	169
8087 Math Support	99	79	89
Run-time Module	250	179	235
C-Link	99	79	89
Btrieve Interface	99	69	85

C Support-Systems

Basic-C Library by C Source	PC \$139
C Sharp - well supported, Source, realtime, tasks, state system	MS \$600
C ToolSet - DIFF, xref, source	MS \$ 95
The HAMMER by OES Systems	PC \$179
Lattice Text Utilities	MS \$105
PC LINT - Checker. Amiga \$89	MS \$119
SECURITY LIB - add encrypt to MS C, C86 programs. Source	\$250 PC \$125

C-Screens. Windows. Graphics

C Power Windows by Entelekon	PC \$119
dBASE Graphics for C	PC \$ 79
Curses by Lattice	PC \$109
ESSENTIAL GRAPHICS - fast, fonts, no royalties	PC \$219
GraphiC - new color version	PC \$319
Topview Toolbasket by Lattice	PC \$209
View Manager for C by Blaise	PC \$219
Vitamin C - screen I/O	PC \$139
Windows for C - fast	PC \$159
Windows for Data - validation	PC \$239

Debuggers

Advanced Trace-86 by Morgan	PC \$149
Modify ASM code on fly.	PC \$109
CODESMITH - visual, modify and rewrite Assembler	PC \$139
C SPRITE - data structures	MS \$ 95
DSD87 - windowing, 8087	PC \$269
Periscope I - own 16K	PC \$129
Periscope II - symbolic, "Reset Box," 2 Screen	PC \$249
Pfix-86 Plus Symbolic Debugger by Phoenix - windows	PC \$115
Software Source by Atron - Lattice, MS C, Pascal, Windows single step, 2 screen, log file.	MS \$199

FEATURE

PC Scheme LISP - by TI. Scheme has special, simple, "orthogonal" syntax. Lexical scoping, block structure, call by value, and tail-recursive semantics. Compiler, EMACS-like editor, DOS. Can support debugging graphics, windowing. PC \$ 95

Fortran & Supporting

ACS Time Series	MS \$449
Forlib + by Alpha - graphics and file routines, comm.	MS \$ 59
MACFortran by Microsoft	MAC \$229
MS Fortran link to C	MS \$219
No Limit - Fortran Scientific	PC \$119
PolyFortran - xref, pp, screen	MS \$149
Prospero - '66, reentrant	MS \$349
RM/Fortran - enhanced "IBM Professional Fortran"	MS \$395
Scientific Subroutines - Matrix	MS \$149
Statistician by Alpha	MS \$269
Strings and Things - register, shell	PC \$ 59

Multilanguage Support

BTRIEVE ISAM	MS \$199
BTRIEVE/N-multiuser	MS \$469
CODESIFTER - Profiler.	MS \$109
HALO Graphics - 115+ devices. Animation, engineering, business.	PC \$229
Any MS language, Lattice, C86	PC \$229
PANEL - Create screen with editor, generates code. Full data validation, no royalties.	Xenix \$539, MS \$239
Pfinish Performance Analyzer	PC \$249
PLINK 86-program-independent	MS \$249
PLINK-86 PLUS - incremental	MS \$369
PolyLibrarian by Polytron	MS \$ 85
PVCS Version Control	MS \$359
Rtrieve - Btrieve front end	MS \$ 79
Screen Sculptor - slick, thorough, fast, BASIC, PASCAL.	PC \$ 99
Xtrieve - organize database	MS \$169
ZAP Communications - VT 100,	PC \$ 95
TEK 4010 emulation, file xfer.	MS \$219
ZView - screen generator	MS \$219

Pascal and Supporting

MetaWINDOW - graphics toolkit	PC \$119
Microsoft PASCAL - faster	MS \$109
MICROTEC PASCAL - 5 memory models, "Iterators". 65 bit 8087 strings.	MS \$665
Pascal Tools - strings, screen	PC \$109
Pascal Tools 2 - by Blaise	MS \$ 85
Pfas - Portable Isam	MS \$185
Prospero Pascal - full ISO +	MS \$349
USCD Pascal - native code	MS \$ 69

RECENT DISCOVERY

APT: Active Prolog Tutor - Guides you in building applications interactively. Arity, Borland, Prolog-86 compatible PC \$ 65

Other Languages

APL*PLUS/PC	PC \$ 469
Artek ADA Compiler - DOD standard minus multitasking	PC \$ 895
CLIPPER-dBASE Compiler	MS \$ 449
ED/ASM - 86 by Oliver	PC \$ 85
MacASM - fast	MAC \$ 99
MasterForth - Forth '83	MAC or PC \$ 125
Microsoft MASM - faster	MS \$ 109
Modula 2 by Volition Systems	MS \$ 250
Modula-2/86 Compiler by Logitech w/ 8087 (\$105), 512K (\$149).	PC \$ 65
Pasm - by Phoenix	MS \$ 219
RPG II by Lattice	PC \$ 639
SNOBOL4+ - great for strings	MS \$ 85
Turbo Edit/ASM - by Speedware	PC \$ 85

Xenix-86 & Supporting

Basic - by Microsoft	\$ 279
Cobol - by Microsoft	\$ 795
Fortran - by Microsoft	\$ 399
PANEL Screen LIB - multi-language	\$ 539
Xenix Complete Development System	\$1149

Other Products

BSW Make - like UNIX make	MS \$ 85
Dan Bricklin's Demo Program	PC \$ 69
dBrief - Customize BRIEF for dBASE development. with BRIEF \$275.	PC \$ 95
H Test/H Format - XT Fix	PC \$ 89
Interactive Easyflow-HavenTree	PC \$ 139
Link & Locate - MSDOS tools to work with Intel and Tektronix projects.	MS \$ 329
LMK - like UNIX make	PC \$ 149
Microsoft Windows	PC \$ 75
Opt Tech Sort - sort, merge	MS \$ 119
PMaker - by Phoenix	PC \$ 139
Polymake by Polytron	MS \$ 79
PolyWindows Dev. Kit	PC \$ 149
PS MAKE	MS \$ 129
Qwik Net - critical path, 125 tasks/ resources, thorough; usable	PC \$ 316
SECRET DISK by Lattice	PC \$ 99
Shrink/Shrinkem - put more files on disk	PC \$ 150
SoftEst - Manage projects.	MS \$ 350
Synergy - Create user interfaces	MS \$ 375
Texsys - control source	MS \$ 89
Visible Computer: 8088 - Simulates demos or any .exe. com. Debugger. 350 pg. tutorial, unprotected	PC \$ 75

Note: All prices subject to change without notice.

Mention this ad. Some prices are specials. Ask about COD and POs. All formats available.

UPS surface shipping add \$3/item.

Call for a catalog, literature, advice and service you can trust

HOURS

8:30 AM - 8:00 PM EST.

800-421-8006

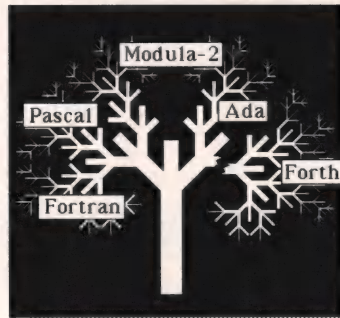
THE PROGRAMMER'S SHOP™

128-D Rockland Street, Hanover, MA 02339
Mass: 800-442-8070 or 617-826-7531 6/86

"The scope and detail of services you provide are exemplary — it's obvious you have given a lot of thought to what information people need . . . For someone like myself, critical appraisals of software and comprehensive collections of offerings such as you have are really useful."

A. Bruce Cyr
Foundation of American College
of Health Care Administrators

Generic Routines in Ada and Modula-2 and Pascal Iterators



In this column I'll look at generic routines in Ada and Modula-2. Ada formally supports generics, whereas Modula-2 provides an indirect approach. I will also examine a new type of extended *for* loop, implemented in a commercial Pascal compiler.

Pascal programmers are familiar with the limitations of strong data typing, which dictates the creation of multiple versions of the same routines to handle different data types. In effect, a lot of effort is spent reinventing the wheel. The situation worsens when arrays are handled by routines such as those for sorting and searching, and for multidimensional arrays, matters are even more complicated. These problems were inherited by the designers of Ada and Modula-2, who decided to put an end to them.

The solution to the above dilemma comes in two stages. The first is the creation of routines that handle different size arrays with the same data types. Many Pascal implementations, as well as standard Modula-2 and Ada, enable programmers to write general-purpose libraries to handle arrays of different sizes. Modula-2 supports one-dimensional open arrays. Some Pascal implementations provide the same feature; others allow multidimensional arrays to be handled. The second stage deals with writing routines that handle different data types and, where appropriate, different array sizes.

Generic Routines in Ada

Consider a short procedure to swap

by Namir Clement
Shammas

two integers, as shown in Listing One, page 110. Without using the generic feature, programmers must have as many versions of *Swap* as

data types encountered. To minimize this waste, you can write a simple generic procedure as shown in Listing Two, page 110. Notice the use of the reserved word *generic* to declare such a routine. In addition, you define the data type *Object* as a *private* type. The generic procedure resembles a template or a mold, and the *private* type represents a blank type.

Generic routines cannot be used directly. Instead, they must first be instantiated (that is, used to create a customized routine). This dictates creating a new, usable routine with a distinct name. In addition the *private* type parameter must be associated with a specific data type. The second step is to write a *use <customized routine name>* statement. To create two versions of *Swap* (one for integers, the other for reals), you can write the following:

```
procedure Swap_Int is new
                                Swap(INTEGER);
use Swap_Int; -- now we are ready!
procedure Swap_Float is new
                                Swap(FLOAT);
use Swap_Float;
```

The above procedures are declared with new names and specify the actual data type to which the customized version is tailored. The keywords *is new* are part of the Ada syntax used in generic instantiation. Operations tackling *private* parameter types are limited to assignments.

Ada permits other kinds of data type parameters. With each class comes a set of operations that are needed to avoid confusing situations

involving generics.

Generic enumerated parameters allow the use of all operators involved with normal enumerated types. This includes relational operators, *SUCC* and *PRED*, and membership operators *in* and *not in*. Generic enumerated types are declared using *< >*. Listing Three, page 110, shows a generic function to return the next element in a circular list. Examples for using the function are shown at the trailing comments: The first uses the enumerated type *Day*, which includes the weekdays; the second example uses an integer subrange to represent hours.

Generic integer parameters allow the use of integer operators in addition to those of the enumerated type. Integer parameters are declared with *range < >*. Listing Four, page 110, shows a generic function that scans an integer-typed array and returns the largest value found. This example also introduces generic arrays. Notice that there are three generic data types: two integers and one array. When the generic function is instantiated, three parameters are needed. The following example produces a function that tackles an array of 100 integers. You are not using the standard type integer specifically, however. Instead you are employing a derived type, called *Whole_Number*, which is a range of integers from 1 to 1,000.

```
type List_Range is range 1..100;
type Whole_Number is integer
                                range 1..1000;
type My_Own_List is array
                                (List_Range) of Whole_Number;
function Highest is new Largest(List
                                _Range, Whole_Number,
                                My_Own_List);
```

When function *Highest* is called, the local variable *Big* is set to 1, the lowest value of the *Whole_Number* type.

Generic floating-point types permit floating-point operations. They are declared using *digits* < >. Listing Five, page 110, shows a generic function that returns the average values of an array of floating-point numbers. Once again there are three generic parameter types, requiring three typed arguments for instantiating the generic function. The following creates a function *Mean* that processes arrays of floating-point numbers, 100 elements long:

```
type List_Range is range 1..100;
type Numbers is array (List_Range)
of FLOAT;
function Mean is new Average(List
_Range, FLOAT, Numbers);
```

Other generic types include the fixed and limited *private*. The generic fixed parameters are declared using *delta* < >. Limited *private* has stricter access rules and is declared using *limited private*.

For the sake of brief listings, I have shown so far examples with one generic routine. This does not reflect any restrictions, though. Multiple generic routines are allowed by Ada—for example, you can write a generic library for complex mathematical operations and functions, and this library can become a template for routines that handle different floating-point types.

Ada extends the types of generic parameters to include subprograms. This feature has two kinds of applications. The first helps to create generic functions with functional parameters. Listing Six, page 110, shows a generic mathematical root finder that uses Newton's method. It also uses a generic floating-point type to make it usable with all floating-point types. The declaration of function *F_of_X* is preceded by the *with* keyword. This tells the compiler that this is a subprogram parameter and not an ordinary generic subprogram. To use the generic procedure *Root*, the client program must define a function that matches *F_of_X*—call it *Cubic_Polynom*. The customized version of *Root* is instantiated for the ordinary *FLOAT* type as shown in the following example:

```
function Newton_Root is new
Root(FLOAT, Cubic_Polynom)
```

The second use of generic subprogram parameters is to provide required operations for generic types that have no predefined operations. Consider, for example, the task of writing a generic shell sorting procedure that potentially handles an array of any data type. As discussed earlier, you must resort to the generic *private* type parameter. This type does not have the much needed predefined inequality operators, however. Ada allows you to overcome this obstacle if you can supply these operators your-

self. Listing Seven, page 110, shows a generic sort program. To instantiate it, you must supply the name of the inequality operator you have written for the data type involved. If it happens to be a standard type, you simply supply the name of the predefined operator. To create a new version for sorting an array of 1,000 integers, for example, you would write the following lines:

```
procedure Sort_Int is new
Shell_Sort(1000,INTEGER,Int_List,">")
```

PolyMake The Leading Make Utility

Are you still using a prehistoric Make? Now, step up to PolyMake, the most powerful and flexible Make utility available for programmers using MS-DOS. PolyMake is like an intelligent assistant that remembers how to rebuild a program when you change one part of the program. PolyMake will automatically invoke your compiler, assembler, linker, librarian or other tools to update a single program or entire software systems when you type — MAKE. PolyMake comes with built-in rules for rebuilding programs but you can also teach it new rules so you don't have to remember the file dependencies in your program. Advanced programmers prefer capabilities like fully recursive makefile processing and the ability to invoke PolyMake with special "flags" and macros that automate a whole range tasks. New Make users appreciate the Step-By-Step tutorial and intuitive commands. Handles source files written in any language. Requires DOS 2.0 & higher. Compatible with LANs, the IBM PC, XT, AT and other MS-DOS PCs. For complete details write for the POLYTRON Programmer's Catalog. **\$99**

PVCS The Most Powerful & Flexible Source Code Revision & Version Control System.

The POLYTRON Version Control System (PVCS) allows programmers, project managers, librarians and system administrators to effectively control the proliferation of revisions and versions of source code in software systems and products. PVCS is a superb tool for programmers and programming teams. (A special LAN version is also available.) If you allow simultaneous changes to a module PVCS can merge the changes into a single new revision. If changes conflict, the user is notified. Powerful capabilities include: Stores and retrieves multiple revisions of text; Maintains a complete history of revisions to act as an "audit trail" to monitor the evolution of a software system; Maintains separate lines of development or "branching"; Provides for levels of security to assure system integrity; Uses an intelligent "difference detection" to minimize the amount of disk space required to store a new version. Requires DOS 2.0 or higher. Compatible with the IBM PC, XT, AT and other MS-DOS PCs. Single User version \$395. 5-station LAN version \$1,000, add \$500 for each additional 5 stations. **\$395**

TO ORDER: VISA/MC 1-800-547-4000, Dept. No. 355; Oregon/Outside US, 503-684-3000
Send Checks/POs To: POLYTRON Corp. 1815 NW 169th Pl., #2110, Dept. No. 355, Beaverton, OR 97006

POLYTRON
High Quality Software Since 1982®

Circle no. 283 on reader service card.

where *Int_List* is a 1,000-integer array type and ">" refers to the predefined operator. Similarly, you can create other versions to sort arrays of records, as long as you supply the needed operator.

Generic Routines in Modula-2

Modula-2 approaches the matter of generic routines very differently

from the way in which Ada does. Modula-2 does not define generic data types and procedures explicitly. Instead, you use its open-array feature, which allows arrays of different sizes but consistent data types to be passed to procedures and functions. The second ingredient is the imported type *WORD*, which represents a unit of data storage. Integers and cardinals are stored in one *WORD*. Reals occupy two *WORDS*. User-defined structures occupy as many *WORDS* as are needed for them

to fit. The bottom line is that Modula-2 can provide the equivalent of Ada's *private* types by using *WORD*-typed open arrays. Cardinals and integers create one small exception to the above—because they occupy one *WORD*, a special but limited class of generic routines can be developed for them.

Listing Eight, page 111, demonstrates a function that searches arrays of cardinals or integers, looking for a match to the supplied key value. The function returns a Boolean value indicating whether the search is successful. The index of the located elements is also returned through the argument-list variable *Index*. Notice that the supplied search value is an integer. This is because array elements are forcibly converted into integers. As a consequence, the function *LinearSearch* can safely handle arrays of integers and cardinals with values ranging from 0 to *MAX(INTEGER)*, which is the common range of values between the two types.

As mentioned earlier, in generic routines in Modula-2, the open array of *WORDS* emulates Ada's *private* types, and I discussed how these types need generic subprogram parameters to provide user-definable operations. Modula-2, however, permits procedural parameter passing. This means you can supply Modula-2 procedures tailored to specific data types. There remains one last ingredient: a sample "template" data type needed when generic arrays are processed.

Listing Nine, page 111, shows a Modula-2 procedure that performs generic shell sorting. The formal argument list includes:

- The data array *L*.
- Two sample template data objects. They are defined as arrays of *WORD* to enable multi-*WORD* types, such as record structures.
- The actual number of data items. This is useful when arrays are partially filled with data.
- The user-definable procedure *Is-Greater* to provide the logical outcome of comparing two data items. This is equivalent to the ">" operator supplied to the Ada generic version. The user function type *UserDefinedProc* is declared as:

```
TYPE UserDefinedProc = PROCEDURE
```

SecretDisk™

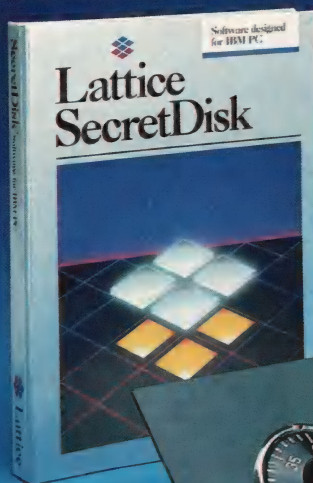
A "...breakthrough for the field of data-security and encryption systems."

—PC Week

PC WEEK REVIEWS said it better than we could. "Until now, these (PC data-security and encryption) systems have been nightmarish creations..."

"With SecretDisk, data security becomes a practical, transparent function—to the point where the processes of creating, manipulating and using encrypted, protected files are indistinguishable from those associated with normal DOS use."

"SecretDisk has the feel of a landmark product. By making data encryption this inexpensive..." How inexpensive? Just \$119.95 to keep your business your business.



Lattice

Available from quality software suppliers or call:
Lattice, Incorporated
Post Office Box 3072
Glen Ellyn, IL 60138
312-858-7950
TWX 910-291-2190



Only
\$119.95



(ARRAY OF WORD, ARRAY OF WORD) :
BOOLEAN;

In comparing the Ada and Modula-2 versions for generic shell sorting, you can see that Modula-2 requires more internal procedures. The procedures *FetchItem* and *PutItem* serve to copy data between the sorted array and the supplied sample object. In the innermost *FOR* loop, two array elements are copied into variables *Sample1* and *Sample2*. The user-supplied *IsGreater* function is used to compare the above data items. If needed, they are swapped simply by writing them back to the original array in a switched order.

The role of user-defined functions is most vital in processing multi-WORD data types. For example, the generic shell sorting procedure might be called to sort an array of records. (See Table 1, below.) The record structure has two fields: one for the name; the other for counting the frequency of occurrence. Now assume you have defined a logical function *GreaterFreq* (see Listing Ten, page 112) to compare two *Frequency* fields of *NameUse* type records. A call to *ShellSort* to arrange a 100-member array in order using the *GreaterFreq* function is written as:

```
ShellSort(NameList, Name1, Name2,  
          100, GreaterFreq)
```

Extended For Loop

I found an interesting extended *for* loop in Professional Pascal (from MetaWare, Santa Cruz, California). Called the iterator, it is composed of two main parts: the iterator declaration and the extended *for* loop. Table 2, right, shows the general syntax of an iterator. It takes two parameter lists: the first is the input list; the second is the output list. Thus the iterator is capable of returning more than one value explicitly, using the *Yield* statement. The latter is directly responsible for passing back results. Table 2 also shows an alternate header declaration for returning a single value, and it shows how the iterator is invoked using the special *for* loop. The loop can have multiple counters and must be matched by the number of arguments in the output parameter list.

Listing Eleven, page 112, shows a

simple example using iterators. The program reads two text files: one contains names; the other, search keys. Neither the names nor the keys are sorted in any order. The iterator *Select* contains two nested loops to match keys with names. When a match is found, it "yields" the indices for the corresponding key and name. This information is used in the extended *for* loop to display the name and the indices for the matching key and name. This shows a feature of iterators: executing the extended *for* loop body only when values are yielded.

Because iterators are separate program bodies, the algorithms they implement can be replaced by more efficient ones, which minimizes the effort associated with the changes. The extended *for* loops need not be altered, although they automatically benefit from any increase in the performance of the iterator bodies. In the case of my search example, the iterator can be modified if either or both

name and key lists are sorted. Putting them in order helps to shorten the search time. The sorting can be carried out by the iterator or, more appropriately, by another program.

Bibliography

Barnes, J. *Programming in Ada*. 2d ed., Reading, Mass.: Addison-Wesley, 1984.

Ford, G., and Wiener, R. *Modula-2: A Software Development Approach*. New York: John Wiley, 1985.

Joyce, E. *Modula-2: A Seafarer's Guide and Shipyard Manual*. Reading, Mass.: Addison-Wesley, 1985.

Young, S. *An Introduction to Ada*. 2d ed. (revised). Chichester, England: Ellis Horwood, 1984.

DDJ

(Listings begin on page 110.)

Vote for your favorite feature/article.
Circle Reader Service No. 4.

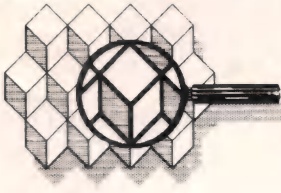
```
TYPE NameUse = RECORD  
    Name : ARRAY [0..29]  
        OF CHAR;  
    Frequency : CARDINAL;  
END;  
  
VAR NameList : ARRAY [0..999] OF  
    NameUse;  
    Name1, Name2 : NameUse;
```

Table 1: Record structure to sort names by frequency of occurrence

```
(* Iterator declaration *)  
iterator <Name>(<Formal parameter list 1>):  
    <Formal parameter list 2>);  
  
(* Declarations here *)  
begin  
    (* Iterator body here *)  
    (* Values returned by a "Yield" statement *)  
    Yield(<Formal parameter list 2>);  
end;  
  
(* Alternate Iterator declaration *)  
iterator <Name>(<Formal parameter list 1>): <type_name>  
  
(* Declarations here *)  
begin  
    (* Iterator body here *)  
    (* Values returned by a "Yield" statement *)  
    Yield(<value>);  
end;  
  
(* Invoking the iterator *)  
for <Counter1>, <Counter2>, . . . , <Counter<n>> in  
    <Iterator_Name>(<Argument list>) do  
    (* Body of loop *)
```

Table 2: The general syntax for an iterator (extended for loop)

OF INTEREST



Unify Corp. has announced an MS-DOS version of the Unify relational database management system software. It includes a full set of development tools, including industry standard SQL and a fourth-generation report writer, and is compatible with Unix. It also provides users with a Host Language Interface to C. Without the Host Language Interface, Unify costs \$995; the Host Language Interface costs \$495.

Interpreters

Version 2.0 of the Run/C Interpreter from **Lifeboat Associates** includes a full-screen editor, improved language implementation, and built-in graphics functions for IBM PC and compatible systems. The built-in editor uses cursor keys and provides block-move and search-and-replace capabilities. Run/C's retail price is \$120.

Pinnacle Systems has announced the port of a fully implemented version of the Unix System V operating system for the Pinnacle XL. Pinnacle's enhancements to the operating system include the implementation of a proprietary memory-management-unit architecture, the structured separation of low-level I/O tasks from operating system functions, and an enhanced version of the serial I/O routines. The retail price is \$12,000.

Version 4.0 of Fortrix-C from **Rapitech Systems** translates FORTRAN state-

ments such as *ASSIGN*, *PAUSE*, *INQUIRE*, and *BACKSPACE*. The program also supports assumed-size, adjustable, and three-dimensional arrays. The program runs on VAX/VMS, Unix, MS-DOS, and other systems on IBM, DEC, AT&T, and Sun computers.

The Santa Cruz Operation has introduced two Unix-based work-alikes of major DOS applications: SCO Professional and SCO FoxBase. SCO Professional offers Lotus 1-2-3 functionality, with fully integrated spreadsheet, database, and graphics. SCO FoxBase can run Ashton-Tate's dBASE II program in multiuser mode without modification. Each program costs \$795.

Version 2.2 of Advanced Trace86 from **Morgan Computing Co.** offers programmers an assembly-language interpreter that also includes a full-screen symbolic debugger. It features BASIC-like commands such as *load*, *save*, *insert*, *delete*, *edit*, *list*, *trace*, and *run*. It can also trace .EXE or .COM programs in disassembled format with labels and variable names scanned from the .MAP file. Advanced Trace86 sells for \$175.

Lattice has introduced an RPG II Compiler for the IBM PC. It supports standard MS-DOS files plus the standard PC keyboard and function keys. It has ISAM files compatible with dBASE III and includes special extensions for string handling. Additional utilities such as Sort/Merge and Source Entry are also available. The RPG II Compiler is priced at \$750.

Clisp is a LISP interpreter from **Westcomp** that is written in the C language. It performs basic LISP func-

tions as well as functions defined in a library, and definitions can be saved to or loaded from disk. Clisp can run under MS-DOS, Version 2, or later. The package costs \$69.95.

Ryan McFarland Corp. has announced the availability of RM/COBOL with Informix-ESQL/COBOL (embedded SQL for COBOL). This capability allows microcomputer RM/COBOL users to access Relational Database Systems' Informix-SQL relational database via SQL statements embedded in their COBOL programs. These programs pass through an RDS translator to convert SQL statements to COBOL prior to compilation; the translated microcomputer COBOL code can then interface with the relational database.

Application Development

Allen, Emerson & Franklin has unveiled Version 2 of the GTP Development System and a Professional Model of GTP. The new version features multiple-screen applications, a context-sensitive help function, a database manager, and global search criteria. It costs \$150.

SofCap has announced H. D. Tuneup, a disk-optimization package for IBM and compatible PCs. H. D. Tuneup eliminates the fragmentation DOS causes in a file system. After a tuneup, every file is internally contiguous and each file is physically adjacent to its directory neighbors. The list price is \$39.95 plus shipping and handling.

Visible Systems Corp. has released The Visible Rules, a productivity tool for software engineering development and mainte-

nance. The Visible Rules software integrated with The Visible Analyst diagramming tool provides a CAD-like software design tool based on Yourdon rules and Gane and Sarson rules. The major features of this software are an enhanced graphics package oriented for data flows, designs examined on both a local and global basis, and a level-to-level balancing feature that verifies the conservation of data flow by comparing a diagram and its lower levels. The Visible Rules software is available for \$595.

Martian Technologies has announced a 16-bit Virtual Disk System for the TurboDOS operating system. The Virtual Disk System features data transfers using the special 80186/80286 block I/O instructions. The average transfer rate is 888,888 bytes per second using an 8-MHz 80186 or more than 1.1 megabytes per second using a 10-MHz CPU. All data reads and writes are checked for hardware parity errors on a sector-by-sector basis. Any parity errors are passed back to TurboDOS for error processing. The pricing for a 2-megabyte SemiDisk with Virtual Disk software is \$1,495; the Virtual Disk Master costs \$3,495.

For the IBM PC/XT

A&T Systems has released DMS/The Disk Management System for IBM PCs and compatibles. DMS is a full-screen, menu-driven software package that allows users to recover lost data, increase system speed and reliability, locate or reorganize information, and execute applications using function

keys. DMS has more than 40 disk- and file-management functions available. It is priced at \$99.

IDEAssociates has introduced Supermax/EMS, a high-performance multi-function board for the IBM PC/AT. It offers expanded, extended, and conventional memory; two serial ports and one parallel port as standard features; and a memory capacity of up to 4 megabytes. The board's expanded-memory feature is compatible with software applications written to the Lotus/Intel/Microsoft Expanded Memory Specification (EMS). The price for Supermax/EMS ranges from \$495 for a bare board to \$2,595 for 4 megabytes of memory.

American Computer & Peripheral's XTsr uses the high-performance, 16-bit, 8088-2 microprocessor and has up to 640K RAM on a four-layer motherboard. Eight sockets provide 64K of usable EPROM for reprogrammable firmware and BIOS. The XTsr has eight expansion slots for memory, up to 2.5 megabytes, and other I/O function cards. Other features include automatic self-test of system components and memory at power-up, MS-DOS 2.11, and Macro-Assembler software and a speaker for audio and music applications. The XTsr is priced at \$2,560.

Aristo Computers has added 640K motherboard upgrades for the IBM PC/XT and the Compaq Portable to its product line. The upgrades are fully compatible with all DOS versions later than 1.xx. They are not suitable for the various XT clones. Retail prices are \$49 for the XT upgrade and \$29 for the Compaq upgrade.

Graphics

Under an agreement with

Graphic Software Systems, IBM has introduced a TopView-compatible graphics application development package, the IBM PC Graphics Development Toolkit 1.1. This package allows programmers to support TopView 1.1 with applications using pop-up menus, windows, icons, and other bit-map images. Applications are compatible with emerging high-performance graphics processor chips through GSS-CGI software and firmware products.

Paradise Systems has introduced a single-chip implementation of the IBM Enhanced Graphics Adaptor standard that supports all the other display standards for the office market. Pega-1 is a single 84-pin integrated circuit that lets OEMs implement the IBM monochrome and color graphics standards (MDA and CGA, respectively), Hercules monochrome graphics, Plantronics ColorPlus, and Paradise color simulation on a monochrome monitor. Development samples with the chip in a board are available for \$200.

Amdek Corp.'s RGB color monitor is compatible with the IBM Professional Graphics Adaptor. It can display 256 colors at a time, selected from a 4,096 color palette. The Color 730 PGA-compatible Analog RGB monitor is available for \$1,099.

Attachmate Corp. has unveiled its 3270 Host Graphics Program, a software package that allows PC users to access host-generated color graphics. The product uses the company's IBM Irma-compatible 3-N-1 Coax Adapter and 3270-PC Emulation Program to manage up to four concurrent host sessions. Any combination of 3270 text or

FTL Modula-II \$49.95!



Your next computer language. The successor to Pascal, Modula is powerful. Why? Once a routine is written, it need never be recompiled. Programs work everywhere from Z80 through VAX.

FTL Modula-II is a full Z80 CP/M compiler (MSDOS version soon)! It's **fast** -- 18K source compiles in 7 seconds! The built-in split screen editor is worth \$60 alone. Some standard features: full recursion, 15 digit reals, CP/M calls, coprocesses, assembler and linker. The one-pass compiler makes true Z80.COM, ROMable code, too. Get the language you've waited for now. Only \$49.95!

FTL Editor Toolkit

Full source to our split-screen programming editor. Curious? Want to customize to your tastes? Want sample Modula-II code? This is perfect for you. Comes with all you need for your personal editor or terminal installer. Just \$39.95!

WORKMAN & ASSOCIATES
1925 East Mountain St.
Pasadena, CA 91104
(818) 791-7979

We have over 200 formats in stock! Please specify your format when ordering. Add \$2.50 per order for shipping. We welcome COD orders!

Circle no. 244 on reader service card.

UNIX/XENIX COMPATIBILITY!



C FUNCTION LIBRARY

C-LIB incorporates over 200 routines to extend the capabilities of C on the IBM PC. Versions are available for the Lattice, Microsoft, and DeSmet (C Ware) C compilers. All libraries are compatible with PC-DOS 2.0+ operating systems.

Featuring:

- UNIX-Compatible "CURSES" Screen/Window Processing Library
- Buffered, Interrupt-driven, Asynchronous Communications Library
- Powerful String Functions
- 8087/Microsoft Floating Point Conversion

C-LIB SOFTWARE AND MANUAL PACKAGE \$195.

Please call or write for additional information.
DEMO DISKETTE AVAILABLE, please include \$5.

VANCE info systems

2818 clay street • san francisco, ca 94115 • (415) 922-6539

IBM & PC-DOS are trademarks of International Business Machines Corp. Lattice is a trademark of Lattice Inc. Xenix & Microsoft C are trademarks of Microsoft Inc. UNIX is a trademark of Bell Labs. C-LIB is a trademark of VANCE info systems.

Circle no. 154 on reader service card.



Power Tools for system builders™

Call today for our free catalog of design aids, compilers, libraries, debuggers, and support tools for Apple and IBM micro computers. The Power Tools catalog includes product descriptions, warranty and license terms, and all the information you need to make an intelligent purchase decision.

TSF offers technical support, competitive pricing, free UPS shipping on orders over \$100, and a reasonable return policy. Visa, MasterCard, and American Express accepted without surcharge. **TSF helps you get your job done.**

Sample Prices:

Microsoft C \$259
MASM 4.0 \$109
Turbo Pascal \$45
Mark Williams C \$375
Lets C \$59
Wendin OS Toolbox \$89
Blaise Async Manager \$137

Call Toll Free
24 hrs a day/7 days a week

Ask For Operator 2053

800-543-6277

Calif: 800-368-7600



TSF

The Software Family™

• Dept. C-2 • 649 Mission Street
• San Francisco • CA 94105
• (415) 957-0111

Circle no. 230 on reader service card.

/* RELAX PROGRAMMERS */

We know how frustrating programming can be... and we decided to do something about it.

We have created an **Environment** — just for programmers (you know who you are), that relieves some of the frustration of programming and increases your productivity.

We called it **KeyFire**. **KeyFire** is an Integrating Programming Environment that combines YOUR editor, compiler, linker, debugger, and make utility into a single entity.

- Reduces the steps of the development cycle (think, edit, compile, link, debug) to single key stroke operations.
- Maintains an **Environment** file of the parameters for each step.
- Switch easily from **Environment** to **Environment**, so you can work on several programs at once — all without leaving **KeyFire**.
- Perform multiple compiles and link in one step.
- Define macros of DOS command sequences.
- Built-in Help utility filled with lots of often needed information (including DOS commands and their options) and you can add more information yourself.
- We know it's great; we know you'll like it; because we constructed **KeyFire** with **KeyFire**!

\$77.00 USA dollars

Ideal for De Smet C. Lattice C and most one or two pass compilers regardless of the language.

For IBM/XT/AT and compatibles, DOS 2.1 or later.

Send check or money order to: **Software Construction**
467 Saratoga Ave. #256
San Jose, CA 95129

OR CALL: 1-800-541-0900
1-800-334-3030 in Calif.

California residents add sales tax. Foreign orders add \$5.
De Smet C is a trademark of Cware Corp.
Lattice C is a trademark of Lattice Inc.



0010 0010
0010 1110
0010 0010

Circle no. 177 on reader service card.

OF INTEREST (continued from page 121)

graphics sessions, as well as a PC application, can be active simultaneously. Both the standard IBM Color Graphics Adaptor and the high-resolution Enhanced Graphics Adaptor are supported. The 3270 Host Graphics Program retails for \$595; the 3-N-1 Coax Adapter retails for \$1,195.

Communications

Network Software Associates has introduced an enhanced version of its AdaptModem, a combination high-speed synchronous modem and Synchronous Data Link Control communications adapter for IBM PC, PC/XT, PC/AT, and compatibles. AdaptModem incorporates the company's Automatic Call Control (ACC) facilities. It includes auto-dial, auto-answer, automatic redial, a 180-entry call directory for speed dialing or unattended automatic operation, automatic phone-line and modem testing procedures, plus modem configuration options. AdaptModem is priced at \$795.

Electronic Specialists'

Kleen Line Security models are available for standard 4-pin telephone modular connectors (FJ-11) and the wider 8-pin connectors (RJ-45). The system uses modern two-stage semiconductor and gas-discharge-tube suppression techniques. An isolated ground is employed to isolate equipment from lightning discharge current. The cost is \$73.95.

Move-It, Version 4, from **Woolf Software Systems** features automatic file compression, keyboard macros, scripting files, XMODEM protocol support, *infilter* and *outfilter* commands, and the ability to send and receive files auto-

matically. Move-It, Version 4, retails for \$150 and runs on the IBM PC/XT/AT and compatibles.

SoftStyle has announced Version 2.1 of its Printworks for Lasers software for IBM PC/XT/AT computers and compatibles. The software offers memory-resident "power printing" typesetting functions and support for three additional laser printers. The new version adds 35 typesetting-like commands that can be activated from a pop-up menu and inserted directly into any text file, using any Epson MX-80 printer-compatible PC application software. Among the 35 laser-printing functions the user can select are 26 built-in laser-printer typefaces, any cartridge in 5 printer font-cartridge slots, 6 downloaded fonts, 26 foreign-language and mathematical symbol sets, 26 point sizes, 6 pitches, 3 line spacings, 4 special shadings, and a variety of type styles. Printworks for Lasers, Version 2.1, is priced at \$125.

Microcom's High Density Modem System (HDMS) offers password protection for both the modem and the operator's consol, as well as dial-back security. HDMS consists of a chassis, a controller, and dual modem cards. Its price ranges from \$1,300 to \$1,500, depending on configuration.

The Netway 100 Dual Session from **Tri-Data** adds dual-session and, in some cases, windowing capabilities to the Netway System, which is designed to link incompatible hosts, terminals, personal computers, and local-area networks. With the Netway System, virtually any ASCII terminal, as well as the AT&T 4540 keyboard display, can be used to access a variety of host computer environ-

ments, including those from IBM, DEC, Sperry Univac, Honeywell, and Data General.

Storage

The Data Vault Subsystems from **Peripheral Technology Corp.** feature mainframe-type Winchester disk drives using the Enhanced Standard Device Interface. These drives offer formatted capacities of 150 to 320 megabytes, data transfer rates of 1.25 megabytes/second, and average access times of less than 20 ms. A microprocessor-equipped intelligent floppy- and hard-disk controller offers full 16-bit direct memory access and simultaneous data transfer. Prices start at \$6,995 for an internal mount 150-megabyte disk and controller and at \$8,495 for an internal 150-megabyte disk, 55-megabyte streaming tape, and controller.

Artificial Intelligence

LISP Machine's ObjectLisp is a portable second-generation object-oriented programming paradigm designed to make the power of object-oriented programming more accessible to programmers at all levels. ObjectLisp features portability, software development, implementation in Common Lisp, and public-domain status.

Miscellaneous

A business information service with data on more than 10 million public and private companies is available from **Dialog Information Services**. Five major categories of applications are available through Dialog Business Connection: corporate intelligence, financial screening, products and markets, sales prospecting, and travel plan-

ning. A start-up package for Dialog Business Connection costs \$145 and includes \$100 worth of free, on-line time for orientation during the first month, a private password, an IBM-compatible communications software disk, and a user's guide with "where-to-find-it" listings.

Advanced Logic Research's Fast/286 is an IBM PC/AT bus- and form-compatible CPU motherboard that features 512K to 2 megabytes on-board memory, an 8-MHz CPU clock, a parallel printer port, and parity checking RAM. It operates under DOS 3.0 or 3.1 and Xenix. The system is priced at \$2,495.

Computer Friends has announced Proteus, a parallel double buffer and data switch. Proteus features a buffer on each of the two output ports, data switch, control manually or via software, and multiple-copy capability on both ports. Its cost is \$199.

The **GMX Micro-20** Single Board Computer combines a 12.5- or 16.67-MHz Motorola MC68020 32-bit microprocessor and an optional MC68881 floating-point coprocessor with 2 megabytes of 32-bit-wide RAM, up to 256K of 32-bit-wide EPROM, four serial ports, an 8-bit parallel port, a 5¼-inch floppy-disk controller, a SASI peripheral interface, and a time-of-day clock with battery backup. A 16-bit expansion connector allows the addition of the off-the-shelf or custom I/O interfaces. The 12.5-MHz version costs \$2,750.

Dwarf Nebula Software has announced 666-Shell, a DOS visual shell that gives users a fully interactive sorted directory, a file browser, a Where Is program, an Egrep program, a Sort routine, and an intelligent disk-copy routine.

ATTENTION

C-PROGRAMMERS

File System Utility Libraries

All products are written entirely in K&R C. Source code included, No Royalties, Powerful & Portable.

BTree Library

75.00

- High speed random and sequential access.
- Multiple keys per data file with up to 16 million records per file.
- Duplicate keys, variable length data records.

ISAM Driver

40.00

- Greatly speeds application development.
- Combines ease of use of database manager with flexibility of programming language.
- Supports multi key files and dynamic index definition.
- Very easy to use.

Make

59.00

- Patterned after the UNIX utility.
- Works for programs written in every language.
- Full macros, File name expansion and built in rules.

Full Documentation and Example Programs Included.

ALL THREE PRODUCTS FOR — **149.00**

For more information call or write:

softfocus

Credit cards accepted.

1343 Stanbury Drive
Oakville, Ontario, Canada
L6L 2J5
(416) 825-0903

Dealer inquiries invited.

Circle no. 259 on reader service card.

Maximum Performance Debugging

DSD86, The PC-DOS Debugger 69.95
DSD87, The PC-DOS Debugger with 8087 Support . 99.95
DSD80, The CP/M Debugger 125.00

 **SoftAdvances**

P.O. Box 49473 • Austin, Texas 78765 • (512) 478-4763

1-800-232-8088

Circle no. 83 on reader service card.

OF INTEREST

(continued from page 123)

The director supports multiple commands at the DOS prompt, full command aliasing, command recall, and full edit at the DOS prompt. The cost is \$39.

Omnitronix is offering a stand-alone, Z80-based, RS-232 microcontroller for commercial applications. The board provides 8K EPROM; one bank of dynamic RAM; and two bidirectional, asynchronous, RS-232 serial ports. The RAM addressing supports 16K, 64K, or 256K DRAM. The controller comes with a UL/CSA-approved wall power supply. The single-piece price for the programmers' kit is \$349; the technical programming pack is available separately for \$14.95.

Harris/Lanier Business Products has announced HarrisDesk, a word-processing package that can be run in a multiuser configuration via the Concept 4300 personal-computer workgroup server or as a stand-alone product on the Harris 2000 PC or any IBM-compatible personal computer. Other features of HarrisDesk include Speller and Task. The Speller function uses a 90,000-word main dictionary and a 6,500-word user-definable dictionary to verify spelling on documents. This function can be performed on screen or run in the background.

Reference Map

A&T Systems Inc., 12904 Olivine Wy., Silver Spring, MD 20904; (301) 384-1425. Reader Service Number 16.

Advanced Logic Research Inc., 2991 E. White Star Ave., Anaheim, CA 92806; (714) 666-2951. Reader Service Number 17.

Allen, Emerson & Frank-

lin Inc., P.O. Box 928, Katy, TX 77492; (713) 391-8570.

Reader Service Number 18.

Amdek Corp., 2201 Lively Blvd., Elk Grove Village, IL 60007; (312) 364-1180. Reader Service Number 19.

American Computer & Peripheral Inc., 2720 Croddy Wy., Santa Ana, CA 92704; (714) 545-2004. Reader Service Number 20.

Aristo Computers, 16811 El Camino Real, Ste. 213, Houston, TX 77058; (800) 3ARISTO, in TX (713) 480-6288. Reader Service Number 21.

Attachmate Corp., 3241 118th St. SE, Bellevue, WA 98005; (206) 644-4010. Reader Service Number 22.

Computer Friends Inc., 6415 S.W. Canyon Ct., Ste. 10, Portland, OR 97221; (503) 297-2321. Reader Service Number 23.

Dialog Information Services Inc., 3460 Hillview Ave., Palo Alto, CA 94304; (415) 858-3742. Reader Service Number 24.

Dwarf Nebula Software, P.O. Box 46, Dept. S, Sugarland, TX 77487. Reader Service Number 25.

Electronic Specialists Inc., 171 S. Main St., Natick, MA 01760; (800) 225-4876, in MA (617) 655-1532. Reader Service Number 26.

GMX Inc., 1337 W. 37th Pl., Chicago, IL 60609; (312) 927-5510. Reader Service Number 27.

Graphic Software Systems Inc., 9590 S.W. Gemini Dr., P.O. Box 4900, Beaverton, OR 97005; (503) 641-2200. Reader Service Number 28.

Harris/Lanier Business Products Inc., 1700 Chantilly Dr. NE, Atlanta, GA 30324; (404) 329-8000. Reader Service Number 29.

IDEAssociates Inc., 35 Dunham Rd., Billerica, MA 01821; (617) 663-6878. Reader Service Number 30.

Lattice Inc., P.O. Box 3072, Glen Ellyn, IL 60138; (312)

858-7950. Reader Service Number 31.

Lifeboat Associates, 55 S. Broadway, Tarrytown, NY 10591 (914) 332-1875. Reader Service Number 32.

LISP Machine Inc., Bldg. 4 Andover Tech Center, 6 Tech Dr., Andover, MA 01810; (617) 682-0500. Reader Service Number 33.

Martian Technologies, 8348 Center Dr., Ste. F, La Mesa, CA 92041; (619) 464-2924. Reader Service Number 34.

Microcom, 1400A Providence Hwy., Norwood, MA 02062; (617) 762-9310. Reader Service Number 35.

Morgan Computing Co., P.O. Box 112730, Carrollton, TX 75011; (214) 245-4763. Reader Service Number 36.

Network Software Associates Inc., 22982 Mill Creek, Laguna Hills, CA 92653; (714) 768-4013. Reader Service Number 37.

Omnitronix Inc., P.O. Box 43, Mercer Island, WA 98040; (206) 236-2983. Reader Service Number 38.

Paradise Systems Inc., 217 E. Grand Ave., South San Francisco, CA 94080; (415) 588-6000. Reader Service Number 39.

Peripheral Technology Corp., 1331 Harlan Ln., Lake Forest, IL 60045; (312) 941-8787. Reader Service Number 40.

Pinnacle Systems Inc., 10355 Brockwood Rd., Dallas, TX 75238; (214) 340-4941. Reader Service Number 41.

Rapitech Systems Inc., Montebello Corporate Park, Suffern, NY 10901; (914) 368-3000. Reader Service Number 42.

Ryan-McFarland Corp., 609 Deep Valley Dr., Rolling Hills Estates, CA 90274; (213) 541-4828. Reader Service Number 43.

Santa Cruz Operation (The), 500 Chestnut St., P.O. Box 1900, Santa Cruz, CA 95061; (408) 425-7222. Reader Service Number 44.

SofCap Inc., P.O. Box 131, Cedar Knolls, NJ 07927; (201) 386-5876. Reader Service Number 45.

SoftStyle Inc., Hawaii Kai Office Building, Ste. 205, 7192 Kalaniana'ole Hwy., Honolulu, HI 96825; (808) 396-6368. Reader Service Number 46.

Tri-Data, 505 E. Middlefield Rd., Mountain View, CA 94043 (415) 969-3700. Reader Service Number 47.

Unify Corp., 4000 Kruse Way Pl., Lake Oswego, OR 97034-2548; (503) 635-6265. Reader Service Number 48.

Visible Systems Corp., 336 Baker Ave., Concord, MA 01742; (617) 369-1800. Reader Service Number 49.

Westcomp, Software Engineering Group, 517 N. Mountain Ave., Upland, CA 91786-5016. Reader Service Number 50.

Woolf Software Systems Inc., 6754 Eton Ave., Canoga Park, CA 91303; (818) 703-8112. Reader Service Number 51.

—Wendelin Colby

DDJ

WIZARD C

MSDOS Edition VERSION 3.0

from Wizard Systems Software, Inc.



Franklin Reynolds '86

Wizard C	\$450.00
ROM Development Package	\$350.00
Combined	\$750.00
Automatic Upgrade Program (annual fee)	\$100.00
Make	\$150.00
Math 87 Library	\$ 50.00



Go With The Winner!!!

(617) 641-2379

DeSmet C

— now with —

LARGE MEMORY

C88 C Compiler*\$109

Full K&R + V7 extensions
Inline **asm**
Assembler, linker, librarian
Full screen editor (SEEtm),
8087 & S/W floating point

C88 with D88 Debugger \$159

Set Breakpoints by line number
or function name
Examine Global/Local variables
by name
Show C source while debugging
Both D88 and User displays

Large Memory \$209 C Compiler

32-Bit Pointers (Full Megabyte
Addressing)
Fast Access of Static Data
Includes D88 Debugger

*Both D88 & Large Memory
available as options at \$50 each.
We also have a C compiler for the
Mac. \$150. Call for details.

— plus —

Graphics \$35
BASICA-like (+src)

Hacker \$25
Source for start-up,
RAM.COM & more

Make \$50
Full UNIX-level

Tools (+source) \$35

XARRAY \$39
Large Arrays (+source)

C WARE CORPORATION

P.O. Box C
SUNNYVALE, CA 94087 USA
(408) 720-9696 TELEX: 358185
Street Address: 505 W. Olive, #767
VISA, MC & AMEX accepted

ADVERTISER INDEX

Reader Service No.	Advertiser	Page No.	Reader Service No.	Advertiser	Page No.
92	Addison Wesley	53	95	MetaWare Incorporated	39
286	Aldebaran Laboratories	47	105	Microprocessors Unlimited	88
178	Archimedes Software, Inc.	113	*	Microsoft Corp.	68-69
*	Ashton-Tate	105	*	Microtec Research	109
277	Aspen Systems	107	*	Mix Software	61
250	Austin Codeworks	104	249	Mortice Kern Systems Inc.	92
182	BC Associates	104	232/		
115	Barrington Systems Inc.	4-5	235	Northwest Instrument Sys.	127
290	Beckemeyer Development Tools	105	243	Norton Utilities	41
159	Blaise Computing	97	137	OES Systems	107
217	Blaise Computing	96	227	Oakland Group, Inc.	17
272	Blast/Communications Res. Group	108	254	Oasys	37
275	Block Island Technology	106	239	PMI	101
161	Borland International	C4	266	Peak Electronics	112
219	Brady Computer Books	106	76	Personal Tex	70
181	C Users Groups	94	221	Plums Hall	71
*	C Ware	126	283	Polytron Corp.	117
237	CompuServe	13	129	Programmer's Connection	63,64,65
94	Consulair Corporation	87	143	Programmers Shop	45
*	Creative Programming	82	141/		
268	Custom Software Systems	108	133	Programmer's Shop	114-115
263	Data Management Consultants	59	295	Proto PC, Inc.	101
203	Datalight	9	206	Raima Corporation	67
258	Desktop AI	83	145	Rational Systems	100
*	Digital Equipment Corp.	21	120	Relational Memory Systems	94
87	Digital Research Computers	91	78	SLR Systems	72
*	DDJ Allen Holub-Shell	78	85	Semi Disk Systems	2
*	DDJ Back Issues	109	83	Soft Advances	123
*	DDJ Bound Volumes	74-75	259	SoftFocus	123
*	DDJ Code Listings	79	177	Software Construction	122
*	DDJ Mac Reprints	113	148/		
*	DDJ C-Products	76-77	152	Solution Systems	56
*	DDJ Toolbook of Forth	73	147	Solution Systems	93
*	DDJ Z80 Toolbook	80	298	Sophco	111
89	Ecosoft, Inc.	29	288	Subject, Wills & Company	49
90	Edward K. Ream	83	*	SwyftWare/Information	
138	Essential Software	40	230	Appliance	88
165	Everost Solutions	34	245/	TSF	122
93	FairCom	113	279*		
*	Gimpel Software	51	108	Tech PC	95
97	Greenleaf Software	25	109	Tecware	7
132	Harvard Softworks	85	222	Tecware	81
274	Hauptpage Computer Works	15	223	Tecware	84
196	Hi Tech Equipment Corp.	94	234	Tecware	89
252	IPT Corporation	92	150	Think Technologies	103
194	InfoPro Systems	19	207	Trio Systems	19
*	Integral Quality	102	256	Turbo Power Software	86
179	Intel Corporation	54-55	154	Van Nostrand Reinhold	83
186	Lahey Computer Systems	98	157	Vance Info Systems	121
101	Lattice, Inc.	118	291	Vermont Creative Software	33
118	Lifeboat	99	269	Visual Age	29
257	Logitech, Inc.	C2	116	Wendin, Inc.	11
187	MacTutor	98	208	Western Wares	88
102	Mark Williams Company	1	244	Wizard Systems	125
199	Media Cybernetics	23	211	Wordtech Systems	C3
				Workman & Associates	121
				Western Computer	57

*This advertiser prefers to be contacted directly; see ad for phone number.

ADVERTISING SALES OFFICES

Midwest
Michele Beaty (317) 875-8093

Southeast
Gary George (404) 897-1923

Northern California/Northwest
Lisa Boudreau (415) 366-3600

Southern California/AZ/NM/TX
Michael Wiener (415) 366-3600

Advertising Director
Robert Horton (415) 366-3600

HOW TO CLEAN UP AFTER YOUR EMULATOR.

The biggest bug of all.

You're developing embedded code for a new microprocessor-based product. Your emulator and logic analyzer have helped you track down all the bugs and fix them. So all you have to do is recompile and release, right?

Wrong.

Because there's no way of knowing if you've actually tested every line of code during real-time execution. Somewhere in that massive program or data space may lurk a time bomb, waiting to blow up and crash your system.

NWIS has you covered.

The SoftAnalyst™ from NWIS gives you precisely the tool you need to release your code with confidence. It's called CodeMap™ and it gives you a fast, simple way to ensure that every statement has been exercised and every data element accessed by your test software.



CodeMap lets you map sixty-three 4K pages anywhere inside a 24-bit address space. Within this window, it will monitor every designated address and automatically report any that aren't accessed.

And you can even go further. Because CodeMap allows you to select status bits, you can determine all variables which have been accessed for read or write operations. Or you can see how much of

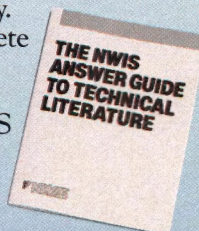
its allotted stack memory space the program has used.

Non-statistical and non-intrusive

All the SoftAnalyst's tools (including CodeMap, Performance Analysis, and Symbolic Trace) are non-statistical, real-time, and non-intrusive, which means you get an absolutely realistic report on your code's behavior in the target system.

We'll show you how:

Call us at 800-547-4445 to schedule a SoftAnalyst demonstration at your facility. Or send for complete product literature and our free publication "The NWIS Answer Guide to Technical Literature."



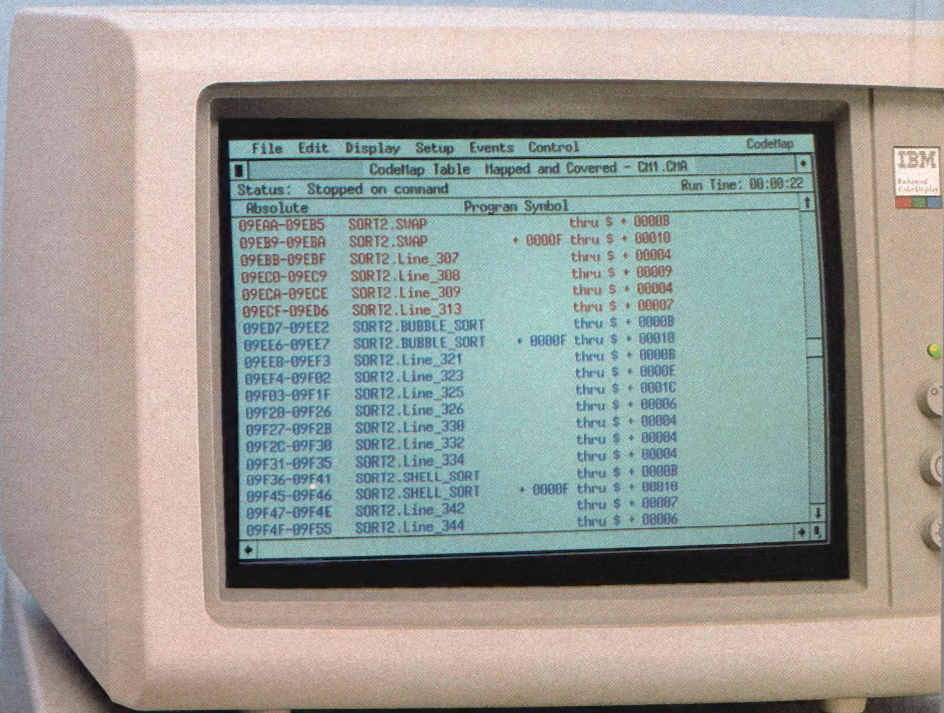
NWIS

NORTHWEST INSTRUMENT SYSTEMS, INC.

P.O. Box 1309 • Beaverton, OR 97075

1-800-547-4445

For Literature circle no. 232 on reader service card.
For Demonstration circle no. 235 on reader service card.



SWAINE'S FLAMES

In Bloomington, Indiana, where I once spent nine years, the humidity and temperature fight it out around ninety at this time of year. In these long, hot, summer days, Bloomington boy John Cougar Mellencamp advises, we need a way to cool ourselves down.

Even for those who didn't grow up in a Mellencamp Midwest, the last weeks of summer can evoke visions of carnivals and lightning bugs. So pop in the American Fool tape, pop the tab off a Leinenkugel, and put your feet up. Here's some light summer reading.

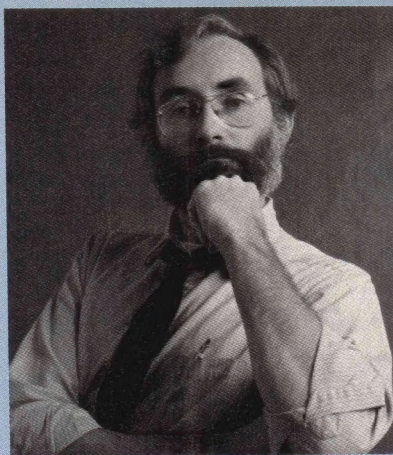
Close enough for rock and roll but inadequate for physics could describe today's programming languages, according to R. Manner.

Except in atmospheric writing like the first paragraph above, comparing temperature and humidity is a no-no. In physics classes we learned that we can meaningfully multiply or divide quantities that involve different units of measurement, but not add, subtract, or compare them. In our computer classes we apparently forgot the lesson, as no computer language today implements these well-known computational rules.

Manner, writing in the March 1986 *SIGPLAN Notices*, proposes changing this. He would extend the syntax of high-level languages so that units such as centimeters, grams, and seconds could be represented as data types, with appropriate compile-time checks on their combination. Units could be defined in terms of other units, as $erg = g \cdot cm \cdot cm / (sec \cdot sec)$, and scale factors could be introduced, as $k = 1000$; $kg = k \cdot g$.

It's an interesting idea that would automate a useful checking technique familiar to every physics student. I recommend the article.

A few American kids doing the best they can are the subjects of *Programmers at Work*, a new book of interviews with programmers published by Microsoft Press.



The interviewer and editors did an excellent job of bringing out the personalities and the ideas. You'll find Kildall and Gates and Bricklin and Frankston here, as well as some less celebrated but thoughtful software artists such as Jaron Lanier. You can recommend this book to your non-programmer friends and still find it worthwhile reading for yourself.

Everyone needs a hand to hold on to, and Carnegie-Mellon University is a little closer to linking up all its students with a network of 3-M workstations (so called because each user will have at least a megabyte of memory, a processing speed of at least one MIPS, and a million-plus pixel display) and a powerful distributed file system called Vice. There are at least three computers that CMU judges capable of functioning as workstations on the system: DEC's MicroVAX, a version of the Apollo Sun, and the IBM RT/PC. Carnegie-Mellon is keeping the system as open as possible to different vendors' advanced workstations.

I can think of two developments that could result from this ambitious project. One: the price of such workstations will come down. CMU officials are predicting a price of around \$3,000 by next fall. Two: at CMU anyway, "Miami Vice" could soon lose rating points to Pittsburgh Vice.

In my weakest moments, I listen to my cousin Corbett's software schemes. His latest is Distributed Semantics.

PROLOG and other logical languages are fine as far as they go, Corbett says.

But, like other languages, all they do is supply a syntax for the expression of ideas; they lack a semantic component. Rules and facts take their meanings from variables that are themselves left undefined.

In expert systems for, say, medical diagnosis, it is critically important that variables have reliable meanings.

Corbett thinks the solution is Distributed Semantics. We just need to be sure that when two programs use the same name, they are referring to the same class of object. People manage this by consulting a dictionary; programs could consult a dictionary of PROLOG-style facts.

Any such dictionary would be inadequate at first; it would have to be able to learn and grow. It would gradually accumulate connotative meanings and would grow into a working lexical model of the world.

Until we have such a dictionary, Corbett suggests, all PROLOG programmers should adopt a uniform commenting style for defining variables. For now, these comments would simply tell the reader what the variable name refers to. When the dictionary becomes available, though, a compiler directive would cause these comments to be interpreted as calls to the dictionary. Thus your comment

```
/* In this program, "horse" means
anything you can put a saddle on. */
```

would become a clause such as

```
horse(X) if env(this program) and
can_wear(X,saddle).
```

with the definitions of *can_wear* and *saddle* to be sought in the dictionary.

Corbett is selling a commenting style manual for \$19.97, prepaid.

Michael Swaine

Michael Swaine
editor-in-chief

When you're hot, you're hot.



Test File Name	HOT C (V.2.33)	Lattice (V.2.15)	Microsoft (V.3.00)
fillscr()	37 sec	61 sec	50 sec
scroll()	45 sec	62 sec	56 sec
prtf()	45 sec	65 sec	58 sec
cpyblk()	62 sec	167 sec	56 sec
cpychr()	38 sec	87 sec	49 sec
diskio()	38 sec	103 sec	41 sec
array()	52 sec	69 sec	77 sec
optimize()	35 sec	32 sec	60 sec
sieve()	93 sec	106 sec	96 sec
rsieve()	54 sec	107 sec	54 sec
minmain()	452 bytes	10260 bytes	1928 bytes
minprtf()	2786 bytes	11684 bytes	5750 bytes
minputs()	1384 bytes	10228 bytes	4296 bytes
minfio()	2694 bytes	10794 bytes	5896 bytes
Price:	\$99	\$500	\$500

HOT C really is.

It's fast. It's powerful. It writes tighter, faster, more efficient code.

And it's cheap. So now just \$99 (suggested retail price) gets you cutting edge performance without the ragged edge problems.

HOT C is an optimizing single-pass compiler. It includes an assembler, linker, debugger, librarian and library source code. It supports large- and

small-memory models and all operators and types except bit fields, with outstanding error and type checking. And in performance, it meets or beats (sometimes dramatically) every other compiler we've come up against.

HOT C is a great power trip for novices, too. A comprehensive tutorial, an impressive text editor and totally revised documentation will have you speaking C like a native in no time flat.

So try HOT C today. Now you can have the steak *and* the sizzle.

For details, just drop a note to WordTech Systems, Inc., P.O. Box 1747, Orinda, CA 94563.

Or call our HOTline at (415) 254-0900.

HOT C

by WORDTECH

HOT C™ WordTech Systems, Inc. Other products trademarked by others.
© WordTech Systems, Inc. 1986.

Circle no. 208 on reader service card.

Borland introduces Turbo Prolog, the natural language of Artificial Intelligence.

Our new Turbo Prolog brings supercomputer power to your IBM® PC and introduces you step-by-step to the fascinating new world of Artificial Intelligence. And does all this for an astounding \$99.95.

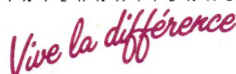


Turbo Prolog radically alters and dramatically improves the brave new world of artificial intelligence—and invites you into that fascinating universe for a humanly intelligent \$99.95.

If you think that this is amazing, you just need to remember that Turbo Prolog is a 5th-generation language—and the kind of language that 21st century computers will use routinely. In fact, you can compare Turbo Prolog to

So don't delay—don't waste a second—get Turbo Prolog now. \$99.95 is an amazingly small price to pay to become an immediate authority—an instant expert on artificial intelligence! The 21st century is only one phone call away.

- ☒ **Debugging:** Complete built-in trace debugging capabilities allowing single stepping of programs.



Circle no. 161 on reader service card.

Other Borland Products include Turbo Pascal, Turbo Tutor, Turbo Lightning, Turbo Database Toolbox, Turbo Graphix Toolbox, Turbo Editor/Tools, Turbo GameWorks, SuperKey, SideKick, SideKick, The Macintosh Office Manager, Reflex, The Analysts, and Traveling SideKick—all of which are registered trademarks or trademarks of Borland International, Inc. or Borland/Analytics, Inc. Turbo Prolog and GeoBase are trademarks and Turbo Pascal is a registered trademark of Borland International, Inc. IBM and AT are registered trademarks of International Business Machines Corp. Copyright 1990 Borland International, Inc. BI-1045D

